2017

Reducing Model Selection Computational Cost by Metamodeling the Evidence

Ramin Madarshahian
University of South Carolina

Reducing Model Selection Computational Cost by Metamodeling the Evidence

by

Ramin Madarshahian

Bachelor of Science
Azad University of Mashhad, 2003
Master of Science
Sharif University of Technology, 2006
Master of Applied Statistics
University of South Carolina, 2012

Submitted in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in

Civil Engineering

College of Engineering and Computing

University of South Carolina

2017

Accepted by:

Juan M. Caicedo, Major Professor

Joseph Flora, Committee Member

Robert L. Mullen, Committee Member

Gabriel A. Terejanu, Committee Member

Cheryl L. Addy, Vice Provost and Dean of the Graduate School

Dedication

I would like to dedicate my dissertation to my beloved parents and my lovely wife.

Acknowledgments

Legal

Partial support was provided by National Science Foundation CAREER: Model Updating Cognitive Systems (MUCogS) project (Award number: 0846258).

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Appreciation

My thanks to the following who are presented in alphabetical order:

- Dr. Juan M. Caicedo, My adviser at the University of South Carolina
- Dr. Joseph Flora, Professor at the University of South Carolina
- Dr. David Hitchcock, Professor at the University of South Carolina
- Dr. Robert L Mullen, Professor and Department Chair at the University of South Carolina
- Dr. Charles Pierce, Professor at the University of South Carolina
- Dr. Gabriel Terejanu, Professor at the University of South Carolina

  and all those who kept me going through your encouragement and support

Abstract

Modeling is an abstraction of reality that lets a designer or engineer perform simulations with a "real" object or structure without testing it physically. However, the result of simulations is as good as their models. Selecting the best model among many candidates is another challenge.

Complex models could translate into more accurate predictions, but they also require larger computational effort. The Bayes factor is often used to compare different models considering the data and analyst's experience. To obtain the Bayes factor, the Bayesian model evidence is calculated; however, calculating the Bayesian model evidence of computationally expensive models is not a straightforward process. The Bayesian evidence is obtained by integrating the multiplication of the likelihood and prior, which is usually performed by numerical methods like Monte Carlo integration algorithms. However, to apply Monte Carlo integration many model evaluations are required, and this is not always feasible when the model is computationally expensive. Metamodeling techniques may be used to reduce the computational cost by replacing the full model with a metamodel. In this research, a different approach to what is proposed in the literature is proposed, and the metamodeling is applied to model the integrand of the Bayesian evidence. Moreover, Probability Distribution Functions, PDFs, are proposed for the metamodel because the integrand of the Bayesian evidence shares some common characteristics with PDFs. For example, they are both always positive. The hypervolume defined by the integrand function is not the same as the hypervolume under the PDF. Therefore to fit the PDF to the integrand, in addition to the PDF parameters, the use of a scale factor is proposed. This scale

factor is used to estimate the Bayesian evidence. To fit the PDFs to the integrand, a scale factor is needed that finally is used to estimate the Bayesian evidence. The proposed method is explained using several examples. It is shown that the number of samples needed is significantly less than the number of samples when Monte Carlo integration methods are used.

Table of Contents

List of Tables

List of Figures

Chapter 1

Background

"Essentially, all models are wrong, but some are useful."

George E. P. Box (1987)

The following material is addressed to readers who are already familiar with fundamentals of probability theory. For those are not familiar with this topic refering to the introductory probability books may be useful [1–4].

Computationally expensive models are used extensively in different fields of science and engineering. Numerical solutions to many model equations such as the Navier-Stokes equations for fluid flows and aerodynamic, heat equations and mass transfer equations for thermodynamics simulations, and finite element analysis for mechanical and structural problems could be very expensive [5–8]. For example, the function, which is used to design an aerodynamic wing, is a solution of the Navier-Stokes equations, and its evaluation may take many hours of CPU time. To complete the design process and wing shape optimization, and to predict the wing performance in different flow scenarios, the function should be evaluated numerous times [9,10]. The problem is more serious when uncertainty quantification (UQ) in design parameters is of interest, which requires many numbers of the function evaluations. Marcov chain Monte Carlo (MCMC) methods are the most common way of doing UQ. These methods are used for sampling a probability distribution of a model's parameters after observing the related data by constructing Markov chains. The quality of obtained distribution is a function of the number of samples. In many cases, to obtain a suitable

distribution, thousands of a model evaluations are required, which is not feasible for long–running models [11–17].

There are alternative models to describe a complex phenomena and to solve scientific or engineering problems. For example, several models, including Landau–Lifshitz–Gilbert, 2D Preisach model and Ising model, are proposed to describe the behavior of magnetic materials [18–21], and all of them demand high computational effort [22]. Another example of computationally expensive models are those used to simulate the delamination in laminated composites. Some of these models which are used to predict delamination growth use energy release rate [23–25]. Some others are developed within the framework of Damage Mechanics [26]. These models are computationally expensive when progressive crack propagation is involved, or when they are used for three-dimensional problems [27–29]. In a paper about prediction of delamination migration in laminated composites, two different modeling techniques are compared [30]. The paper showed that the time needed to perform the extended cohesive damage (ECDM) modeling can be 90% less than the time needed for the standard cohesive zone (CZM) modeling; however, the CPU time is still more than 1000 s, which makes both of these models computationally expensive for uncertainty quantification or Bayesian model selection.

Surrogate modeling, also called metamodeling, is commonly used to reduce the computational cost. The computationally expensive model is replaced by a fast model using a number of samples from the expensive model or function. In this process, only samples are needed for fitting the metamodel, which is usually less than the needed samples for design, optimization, UQ, and model comparison [31].

The objective of this study is to propose an innovative technique to reduce the computational cost of model comparison when evaluation of the models is computationally expensive. The hypothesis is that metamodels can be used to model the evidence and these metamodels can reduce the computational cost to compute the

Bayes factor for model comparison. To address this hypothesis, this research needs a study on Bayesian inference, model selection and metamodeling. In the following sections of this chapter, each of these areas are discussed briefly. In the first section, an introduction to Bayesian inference is presented. This includes a history about the Bayesian modeling and philosophical difference between a Bayesian and frequentist. In the second section, model selection process is discussed, and the Bayes factor is introduced. The last section summarizes the general idea behind the metamodeling and its implementation. Also, this section briefly explains some of the common metamodeling techniques.

1.1 Bayesian inference

For a statistician usually there are two types of reasoning: frequentism and Bayesianism. The way that these two approaches look at probability makes the difference between them. A frequentist obtains the probability of measuring a value by repeating the measurement many times, such that the frequency of an event can express the probability. For a Bayesian probability is the degree of his or her certainty about an event. In other words, the probability is our knowledge and belief about that event [32].

The way that probability is viewed affects the understanding of uncertainty. Uncertainty is generally categorized as either aleatory or epistemic. Aleatoric uncertainty is the intrinsic randomness of a phenomenon, while epistemic uncertainty is from the lack of knowledge. Then, epistemic uncertainty can be reduced by learning more about the system being modeled. Bayesian modeling is a suitable framework to model the epistemic uncertainty because it can consider the prior knowledge about the parameters, and update it based on available data or observations [33].

Bayes theorem was introduced by Thomas Bayes, the 18th century statistician and philosopher.

$$P_Y(y|x) = \frac{P_X(x|y)P_Y(y)}{P_X(x)} \tag{1.1}$$

$P_Y(y|x)$ is the conditional probability distribution function (PDF) for Y given $X = x$, which is obtained from the conditional PDF for X given fixed parameter(s) of y when the investigator has some prior knowledge about Y as reflected in $P_Y(y)$. Bayes' theorem, as it is shown in Equation 1.1, is not the distinction point of frequentism and Bayesianism. $P_Y(y)$ is the initial belief of the investigator about the possibility that an unknown parameter of Y takes the value of y, therefore $P_Y(y)$ is a subjective prior for the conditional PDF of Y given data [34,35]. For example, consider a simply supported concrete beam. If the parameter of interest is the moment of inertia for the cracked cross section, $P_Y(y)$ can be assumed to have a truncated normal distribution which is limited by zero and the moment of inertia of the section before cracking.

$P_Y(y|x)$, or posterior, is the inference about the parameter(s), Y, of a model given taken data X. The mean, median or some quantile intervals for possible value of those parameters, can be obtained from the posterior. The likelihood ($P_X(x|y)$) describes the probability of the sampled data given a specific value of the parameters $Y = y$. The denominator of Equation 1.1 is the probability of data. This is a constant that can be obtained by getting the integral from the denominator over Y. However, it is not a function of Y, and can be considered as a normalizing constant. Therefore, Equation 1.1 is often written as follows:

$$P_Y(y|x) \propto P_X(x|y)P_Y(y) \tag{1.2}$$

Where $\propto$ indicates "proportional to". This new form of the equation is helpful to obtain the posterior using numerical methods. Estimating the posterior is not always a straightforward process. A common strategy is to create samples of the distribution using numerical algorithms [36–38]. A family of MCMC methods, such as Gibbs sampling and Hamiltonian Monte Carlo, is available in the literature and has been

widely implemented for model updating [39,40]. Some authors have proposed modifications to the numerical methods to reduce the computational cost [41,42]. However, some models can still be too computationally expensive even for these sophisticated algorithms. For example, in a comparison study on Metropolis–Hastings proposal algorithms, four algorithms are compared in terms of CPU time to obtain 10000 samples from different target distributions. When the target distribution was a bivariate mixed normal, the CPU time for adaptive rejection Metropolis sampling (ARMS), normal kernel coupler (NKC), adaptive triangular Metropolis sampling (ATRIMS), and adaptive trapezoid Metropolis sampling (ATRAMS) was 10.06 s, 16.51 s, 4.99 s, and 6.04 s, respectively [43]. Although it shows a considerable improvement in CPU time when , for example, the ATRIMS algorithm is selected over NKC, in many cases, this ratio of improvement is not enough to resolve challanges of working with computationally expensive models e.g. a model which its one time evaluation takes several hours.

1.2 Model selection

Growth of computer technology provides the room for trying complex theorems to model scientific and engineering data. Model selection is the task of selecting a model or family of models that best represents the physical behavior of the system given measured data. A mechanical system can be modeled depending on different assumptions, leading to the formulation of different models. For example, several models have been proposed to describe the behavior of a magneto-rheological (MR) damper such as the hyperbolic tangent (HT), Bouc-Wen, Dahl and algebraic models [44].

Model selection become more challenging when a model is supposed to be used for prediction. A model with more parameters may better fit to the data, but sometimes its predictive performance is reduced. The general rule is, among alternative

hypotheses which capture the underlying structure of the data, those with fewer variables are more promising. In model selection, this rule is recognized as Occam's razor, and it is implemented in varieties of model selection method [45]. Akaike Information Criterion (AIC) is one of those methods that considers overfitting in the model selection [46]. AIC is similar to the least squares method when models with the same number of parameters are compared. However, by applying Occam's razor, a penalty is defined for models with more parameters [46–50]. Bayesian Information Criterion (BIC) or Schwarz criterion is closely related to the AIC with a larger penalty on numbers of parameters [51–53]. It was shown that AIC can be obtained based on BIC formulation applying different priors, however there is some research that argues AIC has better practical performance and convergence than BIC [54–56]. Although BIC is defined in a Bayesian framework, the Bayesian model comparison is actually recognized by the use of Bayes factors. Bayesian inference can be used to calculate the probability ratios for representation of a mechanical system. In other words, the Bayes factor is used to evaluate evidence in favor of a null hypothesis. Null and alternative hypotheses can be two competing theories which are described by two different models [57,58]. Furthermore, by using the Bayes factor, it is possible to consider available information, priors, about the models' parameters [59]. The other advantages of using the Bayes factor is its application when one wants to use a combination of models for prediction, and the weights of each model can be obtained considering Bayes factors [60]. Some research suggests values greater than 3.2 are considered substantial evidence against null hypothesis [61], however the threshold is also depends on the context [57].

Bayes' theorem for a model with parameters θ, given data D, can be written as follows [62]:

$$P(\theta|DM) = \frac{P(\theta|M)P(D|\theta M)}{P(D|M)} \tag{1.3}$$

The normalizing constant is the denominator of Equation 1.3 and is called the Bayesian

evidence. The ratio of the Bayesian evidence for two different models is the Bayes factor. Similar to estimating the posterior, calculating the Bayesian evidence for computationally expensive model is not an easy process. Numerical methods should be implemented to calculate a integral of a function which is obtained from product of the likelihood and priors. Using numerical methods requiers taking many samples which is not feasible when the model is computationally expensive. This work focuses on reducing the computational time in this senario. In this research a metamodeling technique is proposed to estimate this integral in less computational effort than sampling-based methods of integration.

The Deviance information criterion (DIC) [63], Likelihood-ratio test [64], and Structural Risk Minimization (SRM) [65] are some of other methods that are used for model selection.

1.3 Metamodeling

Metamodel, which is also known as surrogate model or emulator, is a model of a model. Imagine that two long-running expensive models are used to describe a set of data. It is of interest to study which of them, considering the available data, is more appropriate for our problem. To compare these models, one may want to use one of the model selection methods like calculating the Bayes factor. You need to evaluate the models in different points of the parameter space. Particularly, in the case of using a sample-based integration method to calculate the Bayesian evidence, thousands of function evaluations may be required [66]. Acquiring this number of samples from these models is not feasible.

One solution is to approximate the model by another less computationally demanding model, i.e., metamodeling, or surrogate modeling. Generally, a metamodeling process can be summarized in the flowchart shown in Figure 1.1.

In metamodeling, the goal is minimizing the number of the model evaluations

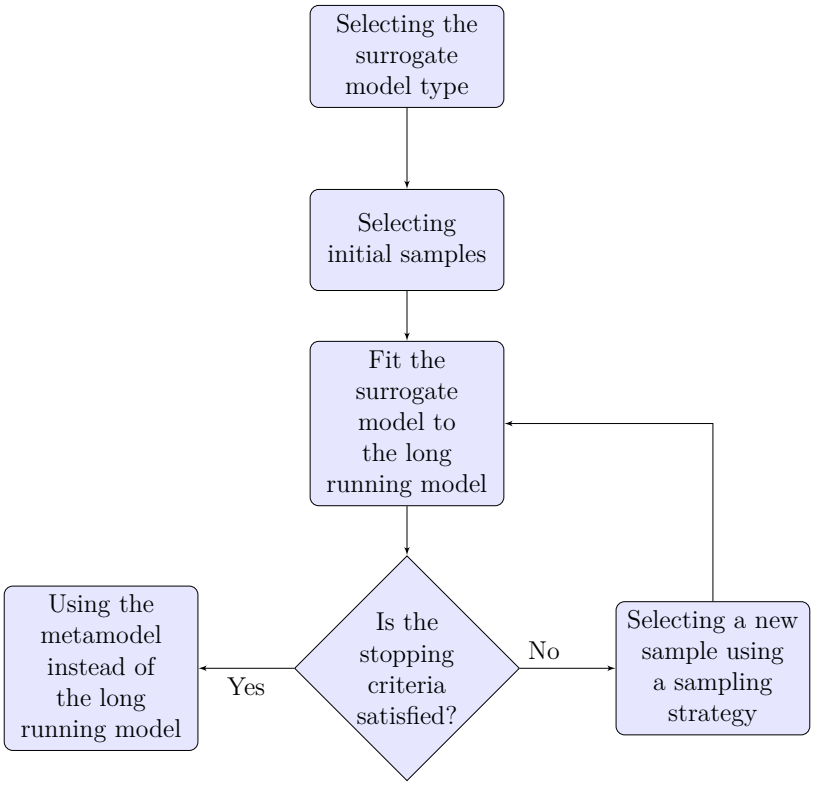
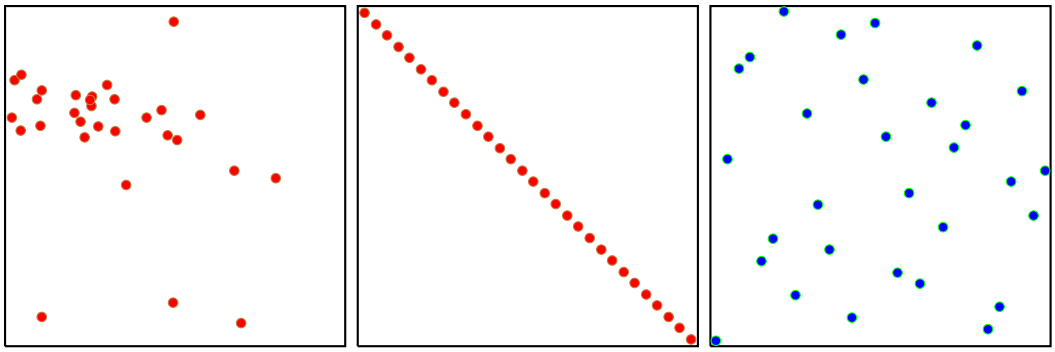


Figure 1.1: General metamodeling process

because running the full model requires a considerable computational effort. To select the samples to fit the metamodel, a sampling plan is needed. A sampling plan has two stages: 1) initial samples and 2) adaptive samples. The initial samples are used for fitting the initial surrogate model. They should be drawn from the design space in a way that their projection onto each variable axis is almost distributed uniformly. This is, it is not ideal if a new sample is selected at the same location of any previous samples as no new information is gained with double the computational cost. Space filling algorithms, including, but not limited, to simple grids, Latin Hypercube [67–69], Hammersley sequence [70], Uniform design [71], and Minimax and Maximin methods [72], are commonly used to generate initial samples. Two poor and one good initial sample set, in a design space for a two variables model, are shown in Figure 1.2. In this example, 30 samples are distributed in the domain of two parameters. In the first sampling scheme on the left subfigure of Figure 1.2,



(Poor) (Poor) (Good)

Figure 1.2: Initial samples with different space filling properties

most of samples are localized in a specific region and there are few samples in other regions. The sampling scheme at the middle could be misleading, when one only looks at the samples' projection in domain of each parameter. However, samples are distributed uniformly in the domain of each parameter, they have been selected highly correlated. This sampling scheme does not provide any information about a large area of the design space. The last sampling scheme shows a good distribution of 30 samples in the domain of parameters. Samples are scattered in all region and provide a good amount of information about the unknown function to form an initial surrogate model.

When the initial surrogate model is built using initial samples, a new sampling plan is required to generate the subsequent samples. The new sampling plan often is an adaptive sampling strategy, because it selects a new sample based on the results obtained from the previously fitted metamodel. The type of adaptive sampling method depends on the type of the surrogate model and the design space properties [73]. To construct the surrogate model, some basic assumptions are frequently made. For example, to use many types of metamodels, it is often assumed that the engineering function is continuous. One can use a less complicated surrogate model if more information is known about the model. For instance, if a surrogate model is used to model the elastic strain and stress relationship of a material, being familiar

with stress and strain relationship of that type of material, a polynomial function, could be a good choice for the surrogate model. So far, many families of metamodels have been proposed in the literature. In the following subsections, some of most common surrogate models are briefly introduced.

Polynomials

Assume that a cantilever beam with a concentrated bending moment at its end. You are told to find the deflection equation of the beam. If you are a structural engineer, you probably assume a second-degree polynomial function with three constants, which are obtained by satisfying the boundary conditions. But in fact, the function used to model the beam deflection is only a polynomial approximation. The true shape of the deflected beam is an arc from a circle because this beam has a constant internal moment along its length and this implies a constant curvature at any point of the beam [74]. A polynomial surrogate model is the classical surrogate model, which is commonly used in response surface model design [75]. A polynomial surrogate model in one dimensional form is shown in Equation 1.4. In this equation, x is a variable and θ contains the metamodel parameters. Equation 1.4 can be extended to more than one dimension by considering more variables, e.g. x_2, x_2^2, x_1x_2, $x_1^2x_2$, etc [76].

$$\hat{y}(x,\theta) = \sum_{i=0}^{n} \theta_i x^i \tag{1.4}$$

Despite its simplicity, polynomial surrogate models are not suitable for the non–linear, multidimensional and multi-modal design spaces. For example, in high dimensional problems many parameters should be considered, and sometimes it is not possible to obtain enough samples to find the higher order polynomials. It should be noted that for a low–modality problem with few numbers of variables, this type of surrogate model is an attractive choice due its simple implementation. Also because of its

simple form, the effect of different variables, including their interactions, can be easily studied. [73].

Moving least-squares

Moving least–squares (MLS), proposed by Lancaster and Salkauskas [77], is a powerful tool for interpolating and approximating design landscapes. It uses weighted least squares measure to construct a continuous function from a set of samples. Weighting of the points is a function of the distance between the observed points and their corresponding predicted values. The weighting function tends to zero at the large value of the distance, and creates a bias towards the points with higher weighting. Furthermore, it would be differentiable if the weighting function is differentiable [78].

Radial basis functions

Radial basis functions (RBF) are functions of distance between a point and a center in the design space. If the distance from center c is shown by $r = |x - c|$, then a RBF is defined as:

$$f(x, c) = f(|x - c|) \tag{1.5}$$

Weighted summation of RBFs is commonly used as a surrogate model to approximate the given functions. Bases are categorized into fixed and parametric functions. Fixed bases are only function of r, such as: linear or $f(r) = r$, cubic or $f(r) = r^3$, and thin plate spline or $f(r) = r^2 \ln r$. Parametric Bases provide more freedom for fitting; however, they are more complex than fixed bases. Gaussian or $f(r) = e^{\frac{-r^2}{2\sigma^2}}$, and multi–quadric or $f(r) = (r^2 + \sigma^2)^{\frac{1}{2}}$, are two examples of parametric bases.

Kriging

Kriging is one of the most common metamodeling techniques. It is from the RBF family of metamodels, which is constructed based on regression against observed values, weighted according to their spatial covariance values. Kriging is an effective metamodeling method due its flexibility to model different shapes of design space [79–81]. When a kriging model is used to approximate a function, $f(x)$, it is constructed from a global model, $\hat{f}(x)$, plus localized departures made by a realization of a stochastic process with mean zero, variance σ, and nonzero covariance as shown in Equation 1.6.

$$f(x) = \hat{f}(x) + \xi(x) \tag{1.6}$$

The term $\hat{f}(x)$, which can be a function similar to the polynomial models, provides a global approximation of the unknown function of interest, $f(x)$, and the term $\xi(x)$ allows the kriging model to interpolate the observed samples by creating localized deviations. The covariance matrix of $\xi(x)$ is a function of correlation between observed samples. For example, in a work which compares kriging with response surface models, authors decided to use a constant for $\hat{f}(x)$ and the Gaussian correlation function shown in Equation 1.7 for the correlation function, $R(x_i, x_j)$.

$$R(x_i, x_j) = e^{-\sum_1^n \theta_k |x_{ik} - x_{jk}|^2} \tag{1.7}$$

where n is number of design variables, and θ_k are the unknown correlation parameters which are used to fit the model, and x_{ik} and x_{jk} are kth component of samples x_i and x_j, respectively [82].

In this section, the general idea of metamodeling was explained, and some of the common metamodels were briefly introduced. Metamodels are generally used to approximate the computationally expensive models. Since many evaluations of these models are not feasible, it is intended to fit the metamodel with minimum numbers of samples. Sampling has two phases: 1) initial sampling 2) adaptive sampling.

Initial samples are used to fit the first metamodel. Filling space algorithms, such as Uniform design and Latin Hypercubes are used to allow samples to distribute over the design space. Adaptive samples are chosen to improve the metamodel fitting and the sampling strategy depends on the type of metamodel and the goal of metamodeling, e.g., optimization, design, and etc.

In this research, it is proposed to use metamodels to directly model the Bayesian evidence instead of approximating the original computationally expensive model, which is used in the structure of the Bayesian inference. In chapter 2, it will be shown that a PDF or a summation of PDFs can be a good candidate for being a metamodel for the Bayesian evidence.

Chapter 2

Metamodeling the evidence

> "An approximate answer to the right question is worth a great deal more than a precise answer to the wrong question."
>
> *John Wilder Tukey*

2.1 introduction

In this work, it is shown how to use metamodels to model the Bayesian evidence and how this allows us to obtain the Bayes factor. This chapter is focused on approximating the Bayesian evidence, while Chapter 3 discusses how to estimate Bayes factors from the metamodels. First, the motivation for this research is explained. Then, since the main application of the Bayesian evidence is model selection, an introduction to model selection is presented. After that, the proposed method is introduced, and finally, an example which was a motivation for this work is shown.

2.2 Motivation

Modeling is an abstraction of reality that lets a designer or engineer perform simulations with a "real" object or structure without testing it physically. However, the result of simulations are as good as their models. Complex models could translate into accurate predictions, but also need more computational effort. The time taken to run computational models of structures can range from a few seconds to weeks

or more, depending on the type of model and the technology used to solve it. The computational time to run a model becomes one of the main challenges in model updating. These techniques require the estimation of the model response with different values of parameters [83–85] in a similar fashion to optimization problems [86]. The computational demands are even higher when model updating is done in a probabilistic fashion [39] and algorithms such as Gibbs sampling [87, 88] are used. Numerous model evaluations are not feasible given the current computational resources available for most researchers and engineers.

Model approximation techniques are very helpful in these computationally expensive scenarios [89, 90]. These techniques develop simplified models of the more complex and computationally expensive model. The first step in most approximation methods is performing the experimental design. Experimental design uses different methods, such as factorial, Latin hypercube, orthogonal arrays, importance sampling, and sequential or adaptive methods to determine the places where the expensive computational model is sampled [91–93]. The design space can be modified based on feedback information from sampling. Sampling methods can help identify regions of the design space where samples would provide additional information and also detect significant variables, including their interaction [94]. The second step is selecting a suitable metamodel to represent the computationally expensive model. The ideal metamodel should be accurate enough to replace the original model with only a fraction of the computational time. Furthermore, the metamodel should be easy to implement [89,94,95]. The fitting process is the next step. The fitting method may be selected depending on the metamodel.

Bayesian model updating is a powerful method to infer the probability of model parameters given some experimental data. MCMC algorithms like Gibbs sampling and the Metropolis-Hasting algorithm are used intensively in these kinds of problems [87, 96]. Importance sampling is a kind of improved MCMC, in which by selecting a

better probability distribution function, the efficiency of the algorithm is improved without sacrificing the accuracy [97]. However, using MCMC methods is not feasible when the model is computationally expensive. Computationally expensive models can be approximated by metamodels, reducing the computational time [98–101].

The question that arises is whether or not metamodeling can be used to calculate the posteriors of parameters rather than the model itself. This was first proposed in a work of Madarshahian and Caicedo in 2015 [102]. Since the posterior is a probability density function, a probability density function multiplied by a scale factor was proposed to use as a metamodel. The preliminary results on [102] found that the scale factor, which is used to fit the proposed surrogate model to the posterior, has the potential to be used to estimate the Bayesian evidence of the model. This work expands those prior findings in a more rigorous way.

Arguably, model selection is one of the main concerns for modeling the complex problems. For example, different finite element models may be suggested to model a large scale structure, and running those models with large number of degrees of freedom, and perhaps nonlinear behavior, needs high computational capabilities. Selecting the best candidate model is even harder to perform because the model needs to be evaluated hundreds or thousands of times.

2.3 Bayesian model selection

Considering the model M_j, and the data D_r, used to infer the model parameters θ, the Bayes theorem is expressed as:

$$P(\theta|D_r M_j) = \frac{P(D_r|\theta M_j)P(\theta|M_j)}{P(D_r|M_j)} \tag{2.1}$$

The marginal probability density function of $P(D_r|M_j)$, which corresponds to the Bayesian evidence for model M_j, can be calculated from the joint distribution of data

and parameters:

$$P(D_r|M_j) = \int_{\Omega_\theta} P(D_r\theta|M_j)d\theta = \int_{\Omega_\theta} P(D_r|\theta M_j)P(\theta|M_j)d\theta \tag{2.2}$$

To investigate which model is the more probable in light of the data, the Bayes theorem can be also used [62]. The probability of M_j given the data is:

$$P(M_j|D_r) = \frac{P(D_r|M_j)P(M_j)}{P(D_r)} \tag{2.3}$$

While this equation is difficult to solve because of the calculation of $P(D_r)$, it is possible to compare the probability ratio between two models M_j and M_k, Bayes factor, using the equation

$$\frac{P(M_j|D_r)}{P(M_k|D_r)} = \frac{P(M_j)}{P(M_k)}\frac{P(D_r|M_j)}{P(D_r|M_k)} \tag{2.4}$$

Notice that $P(D_r|M_j)$ can be calculated as shown in equation 2.3. Assessing the impact of data and considering the prior belief of the analyst about a model and its parameters, computation of Bayes factor is an appealing method for model selection.

2.4 Proposed method

The Bayesian evidence of model M_j, Equation 2.2, is not easy to calculate when the model is computationally expensive. The integral usually is calculated by numerical methods like sampling-based methods of integration, such as Monte carlo methods [87] including Recursive Stratified Sampling [103], VEGAS algorithm [104, 105], and Adaptive Importance Sampling [106]. These methods are often based on the law of large numbers [87, 107] and require a large number of samples to converge. However, in many realistic applications, specially in structural engineering, models are computationally expensive, and can take minutes or hours to run. Prior to explaining the proposed method, it would be useful to define an acronym for the integrand in Equation (2.2). This acronym is inspired from PDF which denotes probability

density function. Similarly, since the answer for the integral is the model's evidence, the integrand is called Evidence Density Function (EDF) in this dissertation. EDF is constructed as the product of several PDFs (likelihood and priors). EDFs have an important charachterestic characteristics of PDFs: they are always positive. In this research, it is proposed to model the EDF with an appropriate PDF. The PDF would be considered as a metamodel or surrogate model and is fitted to the EDF using few model evaluations.

Selection of the type of the PDF depends on the numbers of the parameters and the nature of the problem. Using a PDF as a metamodel has some advantages over other surrogate models [102], for example, like EDFs, PDFs are always positive. Another important advantage of using PDFs as metamodel is obtained from Kolmogorov second axiom, which states that the integral of a PDF over its parameters is always equal to one. The EDF integral is not one, so, a scale factor is needed to fit the metamodel to the EDF. After replacing the EDF by the proposed metamodel, one may calculate the integral of the metamodel to obtain the evidence. The advantage of the proposed method is that the scale factors are not a function of the integral's variables, and can be factored out of the integral. The remaining part is a PDF, which by definition, integrates to one. Therefore, the metamodel's scale factor would be the model's Bayesian evidence.

In some problems, the integration's domain, or domain of the model parameters, is not equal to the domain of the PDF. For example, if a normal distribution function, which is defined from minus to plus infinity, is used to model an EDF, the scale factor would be larger than true Bayesian evidence of the model. In this case, the cumulative distribution function, CDF, of the metamodel can be used in combination with the scale factor to obtain the evidence. Since PDFs used for metamodel are usually selected from common types of PDFs, their CDF equation are readily available and can be directly used without any further integration. Therefore, the EDF is modeled

by a suitable metamodel as follows:

$$P(D_r|\theta)P(\theta) = \rho f(\theta, \beta^*) \tag{2.5}$$

Where $f(\theta, \beta^*)$ represents a PDF, which parameters are represented by β^*, ρ is scale factor, and $\beta = \{\rho, \beta^*\}$ is defined as the metamodel parameters. Then:

$$\int_{\Omega_\theta} \rho f(\theta, \beta^*) d\theta = \rho F|\Omega_\theta \tag{2.6}$$

Where F is the CDF of $f(\theta, \beta^*)$ which is confined by parameter domain Ω_θ. To estimate β^* and ρ, another Bayesian equation could be formed. The samples from the EDF, D_m, are used for the new inference.

$$P(\beta|D_m) = \frac{P(D_m|\beta)P(\beta)}{P(D_m)} \tag{2.7}$$

Let's say $D_m : (X : x_1, x_2, ...x_n, Y : y_1, y_2, ..., y_n)$ such that $x_i \subset \theta$, and $y_i = P(D_r|\theta = x_i)P(\theta = x_i)$. Assuming a normal distribution for the likelihood in eq (2.7), we obtain:

$$P(D_m|\beta, \sigma_{lik}) = \prod_{i=1}^{i=n} \frac{1}{\sqrt{2\sigma_{lik}\pi}} e^{-\frac{(g(x_i,\beta)-y_i)^2}{2\sigma_{lik}^2}} \tag{2.8}$$

In equation 2.8, σ_{lik} is the standard deviation of the likelihood. The posterior describing the metamodel parameters is obtained:

$$P(\beta, \sigma_{lik}|D) = \prod_{i=1}^{i=n} \frac{1}{\sqrt{2\sigma_{lik}\pi}} e^{-\frac{(g(x_i,\beta)-y_i)^2}{2\sigma_{lik}^2}} P(\beta)P(\sigma_{lik}) \tag{2.9}$$

It can be said that the metamodel parameters are obtained in a way that samples from the true EDF scatter around the metamodel following a normal distribution with standard deviation of σ_{lik}, which can provide an estimate for inherent uncertainty in the prediction. Using a normal density function for the likelihood is common, but it should be noticed that, depending on the nature of a problem, other types of distributions may be used for the likelihood as well [108].

Different PDFs can be used depending on the analyst experience with the EDF. One suggestion would be Gaussian Mixture Models (GMM) [109]. In this case, each

of the Gaussian functions has a scale factor, ρ_i, and depending on the domain of the model parameters, the modification factor, F_i, is used to estimate the Bayesian evidence. Finally, the summation of $\rho_i F_i$ yields the model's evidence (Equation 2.6). For example, let's consider a case in which EDF is a function of two independent variables, θ_1 and θ_2. Then, using m numbers of Gaussians, the metamodel is defined as follows:

$$f(\theta, \beta) = \sum_{i=1}^{m} \rho_i \phi(\theta_1 - \mu_{\theta 1i}, \sigma_{\theta 1i}) \phi(\theta_2 - \mu_{\theta 2i}, \sigma_{\theta 2i}) \tag{2.10}$$

In Equation 2.10, ϕ represents a normal distribution function and $\beta = \{\rho_i, \mu_{\theta 1i}, \sigma_{\theta 1i}, \mu_{\theta 2i}, \sigma_{\theta 2i}\}$ for $i = 1, ..., m$ are the metamodel's parameters. This is equivalent to the summation of m bivariate normal distribution with a diagonal covariance matrix. Then, the evidence is obtained by calculating the volume under this metamodel as follows:

$$\begin{aligned} \int_{\theta_{1d}}^{\theta_{1u}} \int_{\theta_{2d}}^{\theta_{2u}} \sum_{i=1}^{m} \rho_i \phi(\theta_1 - \mu_{\theta 1i}, \sigma_{\theta 1i}) \phi(\theta_2 - \mu_{\theta 2i}, \sigma_{\theta 2i}) d\theta_1 d\theta_2 = \\ \sum_{i=1}^{m} \rho_i \Phi(\mu_{\theta 1i}, \sigma_{\theta 1i})|_{\theta_{1d}}^{\theta_{1u}} \Phi(\mu_{\theta 2i}, \sigma_{\theta 2i})_{\theta_{2d}}^{\theta_{2u}} = \sum_{i=1}^{m} \rho_i F_i \end{aligned} \tag{2.11}$$

Where Φ represents a normal cumulative distribution function, and θ_{1id}, and θ_{1iu} are the limits of the integral in θ_{1i}'s domain. In a the same way, θ_{2id} and θ_{2iu} define the interval of integration for the θ_{2i}'s domain. The evidence would be estimated as $\sum_{i=1}^{m} \rho_i$ when $\theta_{1d} = \theta_{2d} = -\infty$, and $\theta_{1u} = \theta_{2u} = \infty$, and in accordance with the second axiom of probability.

MCMC methods can be used to sample the posterior of the parameters of the metamodel. Since the scale factor, ρ, is a random variable, a distribution expressing the uncertainty of the estimation of the Bayesian eveidence can be obtained. This lets us obtain the Bayes factor statistical properties , such as the mean and standard deviation.

In the following section, an example is presented to demonstrate how the method is used to model the posterior. In the next chapter, two more examples are disscussed to illustrate how the method is used to estimate the Bayes factor. It is worth mentioning

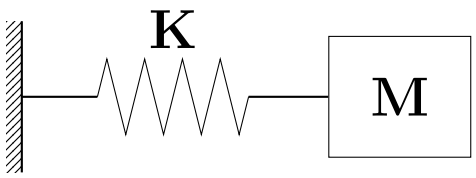


Figure 2.1: Single degree of freedom spring

that, in this work, the focus is on the concept of the proposed method, i.e. the idea of implementing the metamodel after constructing the Bayesian model. Defining a criteria for selecting the appropriate metamodel, sampling strategy and stopping criteria would be topics for future research. The only exception is the sampling strategy proposed in Chapter 4, which was found to perform well with the proposed method.

2.5 Example: Metamodeling the EDF

The single degree of freedom (SDOF) system, shown in figure 2.1, is used to illustrate how remodeling the EDF can reduce the number of evaluations of a structural model. The model is not computationally expensive, and it is used to show how the methodology is implemented. The success of the method will be measured by the number of evaluations of the model, rather that the computational time. This is a model for a structure with unknown stiffness, but measurable natural frequency. So the stiffness (K) is the parameter to be updated, while the mass (M) is assumed deterministic with known value of 100 kg. The "true" distribution for the stiffness is assumed to be normal with a mean of $1000\,\frac{\mathrm{N}}{\mathrm{m}}$ and a standard deviation of $10\,\frac{\mathrm{N}}{\mathrm{m}}$. Ten realizations of the frequencies are theoretically obtained and considered as experimental data. These frequencies are shown in table 2.1.

First, the posterior PDF of K is estimated using a traditional Bayesian inference without the use of the metamodel. Then, a normal distribution is used to re-model the EDF. It should be noted that the EDF and the posterior are proportional. It

Table 2.1: Simulated experimental natural frequencies ω_e

Test No.	$\omega_e \frac{\text{Rad}}{\text{s}}$	Test No.	$\omega_e \frac{\text{Rad}}{\text{s}}$
#1	3.2058	#6	3.1673
#2	3.2184	#7	3.1759
#3	3.1677	#8	3.1264
#4	3.1554	#9	3.1911
#5	3.1415	#10	3.1708

means obtaining the EDF, one can obtain the statistical properties of the posterior as well. The python package PyMC is used to sample the posterior PDFs [110].

Inference without using the metamodel

The natural frequency of a SDOF with small damping ratios can be estimated using the equation:

$$\omega_t = \sqrt{\frac{K}{M}} \tag{2.12}$$

The likelihood is assumed to be Gaussian and it is calculated as follows:

$$P(\omega_e|K) = \prod_{i=1}^{n} \frac{1}{\sigma_{lik}\sqrt{2\pi}} e^{-\frac{(\omega_t - \omega_{e_i})^2}{2\sigma_{lik}^2}} \tag{2.13}$$

In this equation n is the number of experimental frequencies (ω_e), and σ_{lik} is the standard deviation of the likelihood function. The value of σ_{lik} is considered deterministic for simplicity but it can be set as a free parameter in the updated process as shown in a future example (Section 4.3). This analysis assumes a Gaussian prior PDF with $\mu_K = 800\,\frac{\text{N}}{\text{m}}$ and $\sigma_K = 80\,\frac{\text{N}}{\text{m}}$. Also since K cannot take negative values, a condition is added to make the probability zero for any negative values. Figure 2.2 shows the posterior function for the case when $\sigma = 0.1\,\text{Hz}$. The value of maximum probability density given the experimental data shown in Table 2.1 is $K = 994.10\,\frac{\text{N}}{\text{m}}$

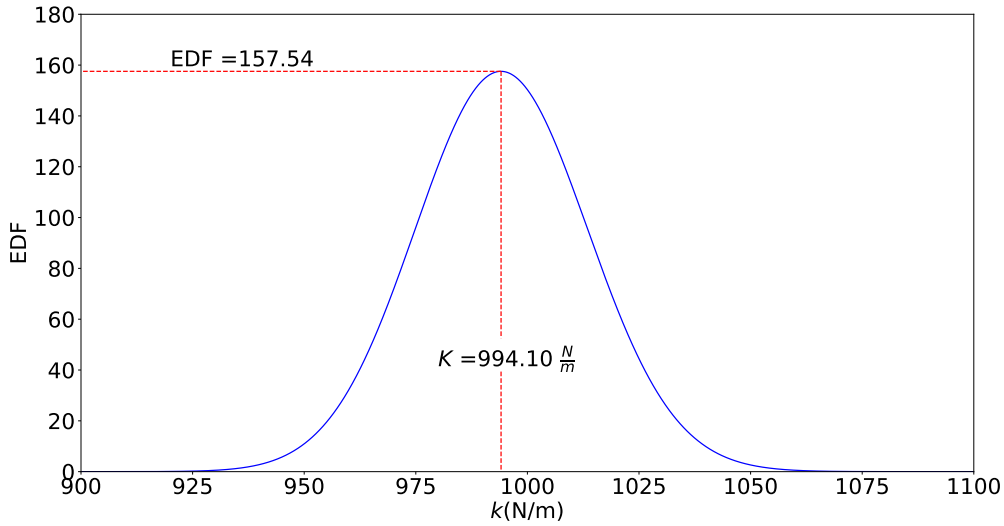


Figure 2.2: Posterior of the stiffness

A MCMC sampling method is implemented to generate a chain which represents the distribution of K.

A total of 20000 samples with a thinning value (MCMC sampling interval) equal to 2 are generated and the first 5000 are discarded. Figure 2.3 shows the histogram of the obtained samples. As expected, the histogram obtained from the samples has the same shape as the posterior function in Figure 2.2. A mean value of $\mu_K = 993.91\,\frac{\mathrm{N}}{\mathrm{m}}$ is obtained. Also the interval with the 95% Highest Posterior Density (HPD) is obtained as $[956.5\,\frac{\mathrm{N}}{\mathrm{m}}\ 1033.1\,\frac{\mathrm{N}}{\mathrm{m}}]$. The theoretical mean value of K is $1000\,\frac{\mathrm{N}}{\mathrm{m}}$ and the value of the highest posterior is $K = 994.1\,\frac{\mathrm{N}}{\mathrm{m}}$. Both are located in the HPD interval. Finally the standard deviation of the samples is calculated as $19.80\,\frac{\mathrm{N}}{\mathrm{m}}$.

Inference metamodeling the EDF

Since the EDF is only a function of one parameter, the metamodel is considered as follows:

$$f(K) = \rho\phi(\beta_1, \beta_2) \tag{2.14}$$

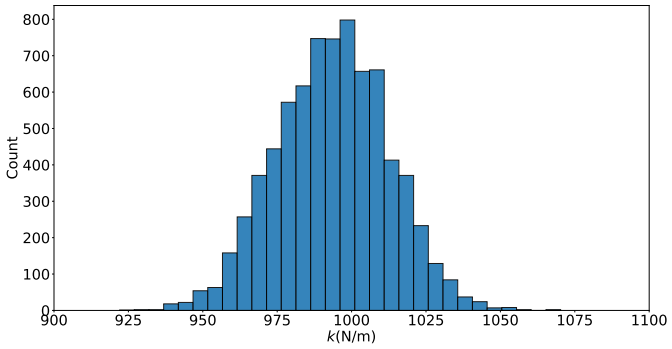


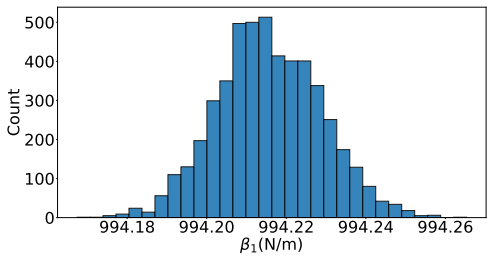
Figure 2.3: Histogram for posterior of the stiffness

In this equation, ρ, β_1, and β_2 are the scale factor, mean and standard deviation of the metamodel, respectively. A few points of the EDF are sampled, and shown in Table 2.2, are selected to use as the data for generating the metamodel. For this example, samples are selected with a constant step in the domain of the parameter. However, usually the peak in the EDF is narrow, and using this kind of sampling is not recommended because the area of high probability might not be sampled.

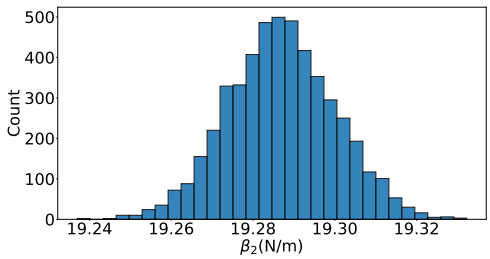
Table 2.2: Ten points of the EDF

$K\ \frac{\mathrm{N}}{\mathrm{m}}$	EDF	$K\ \frac{\mathrm{N}}{\mathrm{m}}$	EDF
910	0.008	1010	112.386
930	0.532	1030	28.667
950	10.951	1050	2.631
970	71.643	1070	0.089
990	154.052	1090	0.001

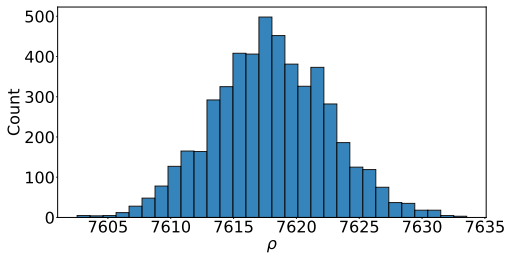
Bayesian inference is used to estimate the probabilities of ρ, β_1, and β_2. The prior for ρ is selected as a normal distribution with the mean value of 1000 and standard deviation of 1000. A normal distribution with $\mu = 800\,\frac{\mathrm{N}}{\mathrm{m}}$ and $\sigma = 80\,\frac{\mathrm{N}}{\mathrm{m}}$ is chosen as the prior for β_1. A normal distribution with $\mu = 300\,\frac{\mathrm{N}}{\mathrm{m}}$ and $\sigma = 100$ is selected for β_2. Gibbs sampling is implemented to generate samples on the parameters of the metamodel. A total of 40000 samples are obtained, 20000 of them are considered burning samples and discarded. Thining value of 4 is used to thin out the chains, and the



(a) Metamodel mean (β_1)



(b) Metamodel standard deviation (β_2)



(c) Metamodel scale factor (ρ)

Figure 2.4: Inference about the metamodel parameters

chains of the parameters are checked for convergence. The mean value for β_1, β_2, and ρ are obtained as 994.21 $\frac{\text{N}}{\text{m}}$, 19.29 $\frac{\text{N}}{\text{m}}$, and 7618.1, respectively. The 95% HPD interval for these parameters as [994.18 $\frac{\text{N}}{\text{m}}$,994.24 $\frac{\text{N}}{\text{m}}$],[19.26 $\frac{\text{N}}{\text{m}}$,19.31 $\frac{\text{N}}{\text{m}}$], and [7609.8 7627.6], respectively. Figure 2.4 shows the histograms of samples obtained for these parameters. Figure 2.5 shows the metamodel obtained from mean values on the analytical EDF.

Concluding remarks

Figure 2.5 shows the metamodel and the EDF and samples taken from the EDF to construct the metamodel. Results show that the metamodel does a good job describing the EDF. The example shows how this technique can reduce the computational cost in computationally expensive models. The traditional Bayesian inference results shown in Figure 2.3 takes 20000 model evaluations, but using the proposed technique only ten evaluations are needed. Using the metamodel one can generate samples for the parameter of interest. This can be considered as a great advan-

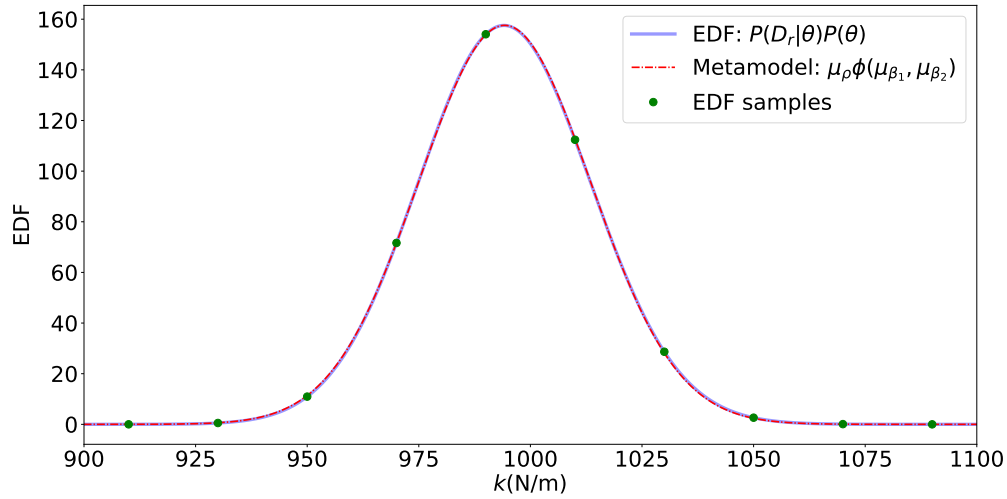


Figure 2.5: Comparison of the EDF and the metamodel

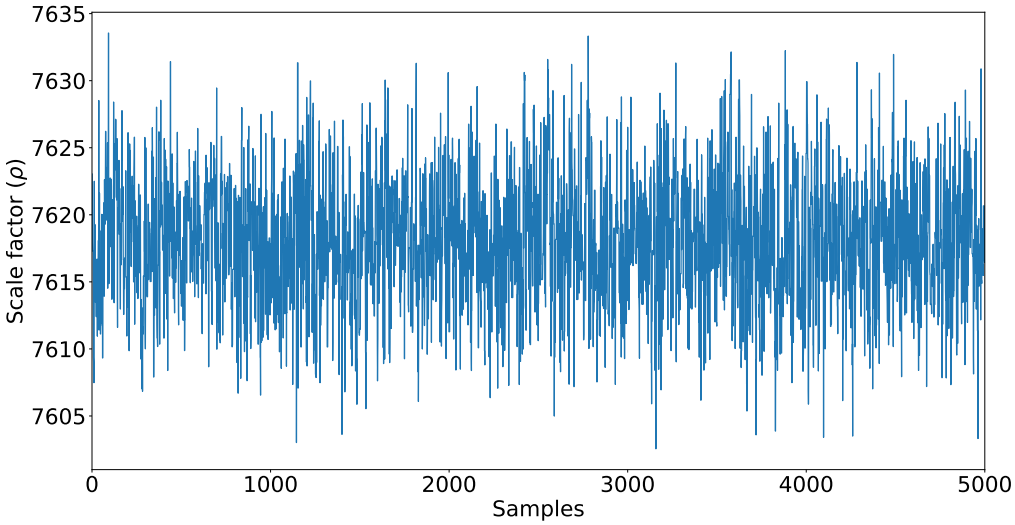


Figure 2.6: Scale factor MCMC chain in example 1

tage when using expensive computational models. The 95 % HPD intervals for K with the proposed approach is [956.5 $\frac{\mathrm{N}}{\mathrm{m}}$,1033.1 $\frac{\mathrm{N}}{\mathrm{m}}$] which is comparable with 95 % HPD interval obtained from original method using the mean value of β_1 and β_2 (i.e. $\mu_{\beta_1} \pm 1.96\ \mu_{\beta_2}$: [956.41$\frac{N}{m}$,1032.02$\frac{N}{m}$]).

Figure 2.6 shows the Markov chain Monte Carlo chain for the scale factor. Since

the integral of a probability distribution over its domain is one based on the second probability axiom, this scale factor is an approximation of the Bayesian evidence of the model. In other words, the method not only provides an estimation for the Bayesian evidence of the model, but also provides us an estimation for its uncertainty. This is used to generate Bayes factor as discussed in the following chapter.

Chapter 3

Metamodeling the Bayes factor

"Those who ignore Statistics
are condemned to reinvent it."

Bradley Efron

3.1 Introduction

This chapter complements the previous chapter by constructing the Bayes factor using the metamodel of the EDF. An example is presented to illustrate the application of the methodology [111]. The example compares two models with one parameter to describe samples obtained from a half circle. Standard deviations of the likelihoods are considered fixed values to simplify the visualization of EDFs. The Bayes factor is calculated implementing the proposed method.

3.2 Metamodeling the Bayes factor

As discussed in Chapter 2, a PDF or a summation of PDFs could be used to model the Bayesian evidence. Imagine there are two computationally expensive models that have been proposed to model a particular phenomena. In addition, data from this phenomena is available. The Bayes factor can be used to select the most probable model in light of the collected data. Using the proposed metamodeling technique, the Bayesian evidence, which is approximated using a metamodel as shown in Equation 2.6, can be used to Estimate the Bayes factor. Recalling Equation 2.6, the

evidence for model i is estimated by $\rho_i F_i|\Omega_{\theta_i}$. Then, the Bayes factor for comparing model i and j is obtained as follows:

$$B_{ij} = \frac{\rho_i F_i|\Omega_{\theta_i}}{\rho_j F_j|\Omega_{\theta_j}} \tag{3.1}$$

The distribution of the estimate of the Bayesian evidence can be sampled using MCMC chains of the metamodel parameters. In other words, it is possible to estimate the statistical properties of the Bayes factor using the chains of the Bayesian evidence.

3.3 Example: Comparing two regression models

In this example, a regression problem is used to demonstrate the capabilities of the proposed method. A few samples are collected from the half-circle function shown in Equation 3.2 and it is considered the "true" model. The half-circle's radius is a random variable with a mean value of 1 m and a standard deviation of 0.01 m. The center of the circle is at coordinates (0 m, 0 m) in $x - y$ plane. Data used for fitting and model selection is collected at $x = \{-1\,\text{m}, -0.5\,\text{m}, 0\,\text{m}, 0.5\,\text{m}, 1\,\text{m}\}$.

$$y = \sqrt{|N(1., 0.01) - x^2|} \tag{3.2}$$

Two candidate models are used to describe the data

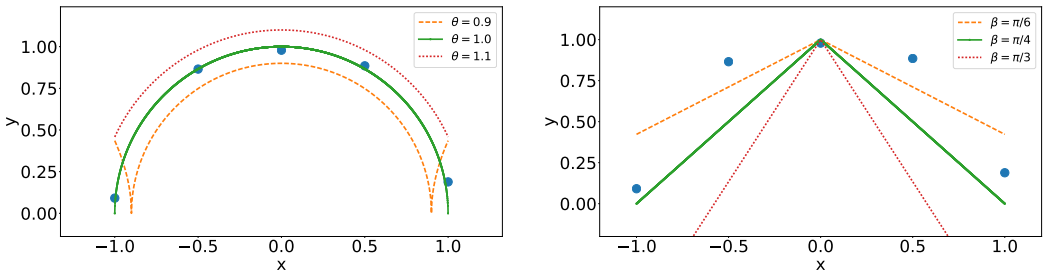
1) half-circle:

$$M_{half-circle} : \sqrt{\theta - x^2} \tag{3.3a}$$

2) chevron:

$$M_{chevron} : 1 - tan(\beta)|x| \tag{3.3b}$$

These models have only one parameter: θ for the half-circle model, and β for the chevron. The data and the two proposed models, using three different values for the model parameter, are shown in Figure 3.1.



(a) half-circle model (b) chevron model

Figure 3.1: Samples and the two proposed regression models

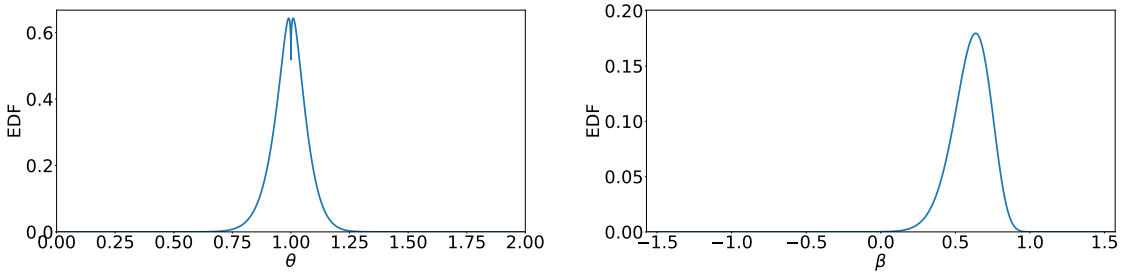
Obtaining the Bayes factor using classical Bayesian inference

The evidence of each model should be calculated to estimate the Bayes factor. In this section, the model evidence of these two models is calculated using Monte Carlo integration. The prior for the half-circle model's parameter is selected as a uniform distribution between 0 m and 2 m. 0 m is selected as a lower limit because the radius can never be a negative number, and 2 m is selected as a upper limit for the radius. A uniform distribution is selected to assume that no additional information is available for this parameter. Notice that the value of the evidence changes depending on the prior of the parameters. However, the method works with any type of priors. In the second model, the slope of lines is shown by $tan(\beta)$. This is because a uniform distribution for the slope would favor steeper lines. This can be shown by plotting several lines while changing the slope by a constant quantity. Numbers from zero to one would cover lines between zero and 45 degrees with respect to the horizon. Numbers between one and infinity would cover angles between 45 and 90 degrees. To solve this issue the model is written in terms of tan(β), then a uniform distribution between $\frac{-\pi}{2}$ and $\frac{\pi}{2}$ is used for β [32]. The evidence for both models are calculated as follows:

$$\int_0^2 (\frac{1}{\sqrt{2\pi\sigma_{lik}^2}})^5 e^{\frac{-1}{2\sigma_{lik}^2}\sum_1^5 (M_{half-circle}(\theta)-y_i)} (\frac{1}{2})^5 d\theta \tag{3.4a}$$

$$\int_{\frac{-\pi}{2}}^{\frac{\pi}{2}} (\frac{1}{\sqrt{2\pi\sigma_{lik}^2}})^5 e^{\frac{-1}{2\sigma_{lik}^2}\sum_1^5 (M_{chevron}(\beta)-y_i)} (\frac{1}{\pi})^5 d\beta \tag{3.4b}$$

Figure 3.2 shows the EDF for both models where the standard deviation of the likelihood (σ_{lik}) is set to 0.3.



(a) Half-circle (b) Chevron

Figure 3.2: EDF of the two proposed regression models

The evidence is calculated by implementing Monte Carlo integration using the Vegas algorithm [105, 112]. To calculate the integral, 500 samples were used. The evidence of the half-circle and the chevron are obtained as 0.1023 ± 0.0008, and 0.0558 ± 0003, respectively. Therefore, the Bayes factor is obtained as follows:

$$B_{12} = \frac{P(D|M_{half\ circle})}{P(D|M_{chevron})} = \frac{0.1023}{0.0558} = 1.833 \tag{3.5}$$

Applying the Proposed Method

As discussed in Chapter 2, the main advantage of using a PDF as a metamodel of the EDF comes from the second axiom of probability that states that the integral of a PDF in its domain is equal to one [113]. Therefore, when the metamodel is fitted to the EDF, the scale factor discussed in section 2.4 would be an approximation of the area under the EDF curve, or the model evidence. Here a normal distribution is used as the metamodel. The equation of the metamodel for the half-circle model is:

$$G_1(\theta) = a_0 g_1(a_1, a_2) = a_0 \frac{1}{\sqrt{2\pi a_2^2}} e^{\frac{-1}{2a_2^2}(\theta - a_1)} \tag{3.6}$$

Here a_0 is the scale factor and a_1 and a_2 are the mean and standard deviation of normal distribution function g_1, respectively. The equation of the metamodel for the chevron model is:

$$G_2(\beta) = b_0 g_1(b_1, b_2) = b_0 \frac{1}{\sqrt{2\pi b_2^2}} e^{\frac{-1}{2b_2^2}(\theta - b_1)} \tag{3.7}$$

Where b_0, b_1 and b_2 are the scale factor, mean and standard deviation, respectively. Some samples should be taken from each EDF to use as the data needed to estimate the distributions of the metamodel's parameters. Here, the stopping criteria is defined as follows:

1) half-circle:

$$\frac{|\mu_{a_{0_i}} - \mu_{a_{0_{i-1}}}|}{\mu_{a_{0_i}}} < 0.02 \tag{3.8a}$$

2) chevron:

$$\frac{|\mu_{b_{0_i}} - \mu_{b_{0_{i-1}}}|}{\mu_{b_{0_i}}} < 0.02 \tag{3.8b}$$

$|\ldots|$ is used to denote the absolute value of the scale factor means difference at iterations i and $i-1$. Sampling would be done in several stages until the stopping criteria is satisfied. Initial samples are sampled from the prior, e.g., 5 samples from prior of θ and β for the half-circle model and the chevron model. Parameters of the metamodel are obtained using these samples. The obtained metamodel usually is not the best fit for the EDF, but it can be used to select the adaptive samples. Here, 3 adaptive samples are sampled from a normal distribution which its parameters are obtained from the previous run. For example, for the half-circle model, MCMC chains of a_1 and a_2 are obtained. Mean values of them are used to define a normal distribution function for sampling. 3 new samples are drawn from this distribution function and are added to the previous samples. Then, the metamodel is updated and the stopping criteria is checked. This process is continued until the stopping criteria is satisfied. Figures 3.3 and 3.4 show the evolution of each metamodel for each iteration. Green diamond markers show the first five samples. The second three adaptive samples

are shown by red circles. This second set of samples are from a narrower region of the model parameter as they are drawn with the help of the updated metamodel. The stopping criteria is satisfied after three iterations for half-circle model and four iterations for chevron model. Only 11 samples for half-circle model and 14 samples for chevron model are needed to fit the metamodel.

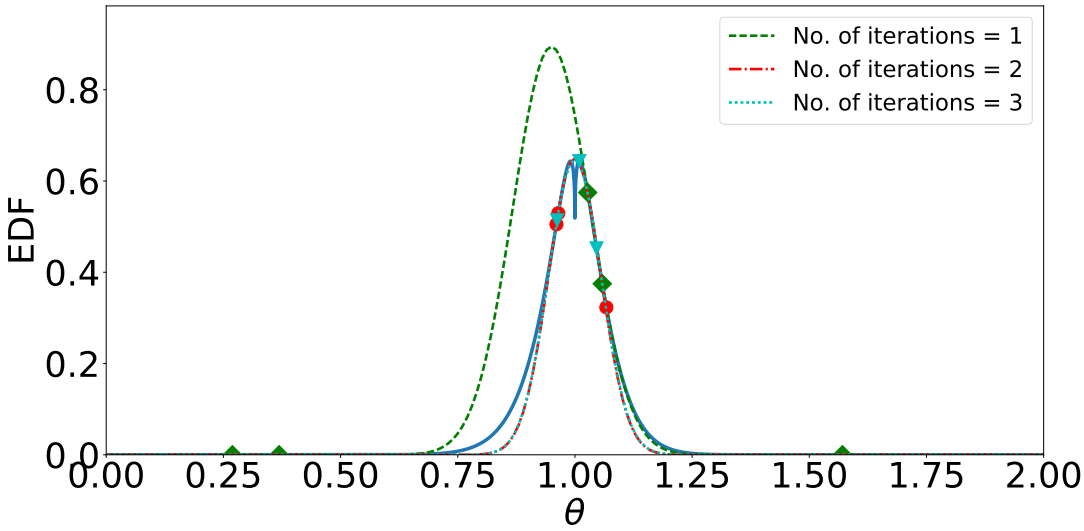


Figure 3.3: Metamodel updating: half-circle

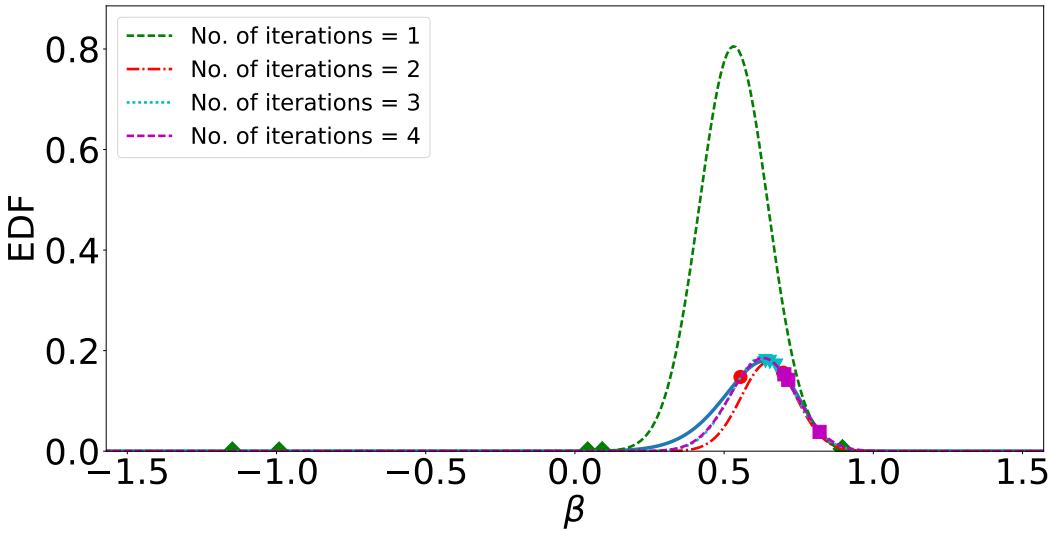


Figure 3.4: Metamodel updating: chevron

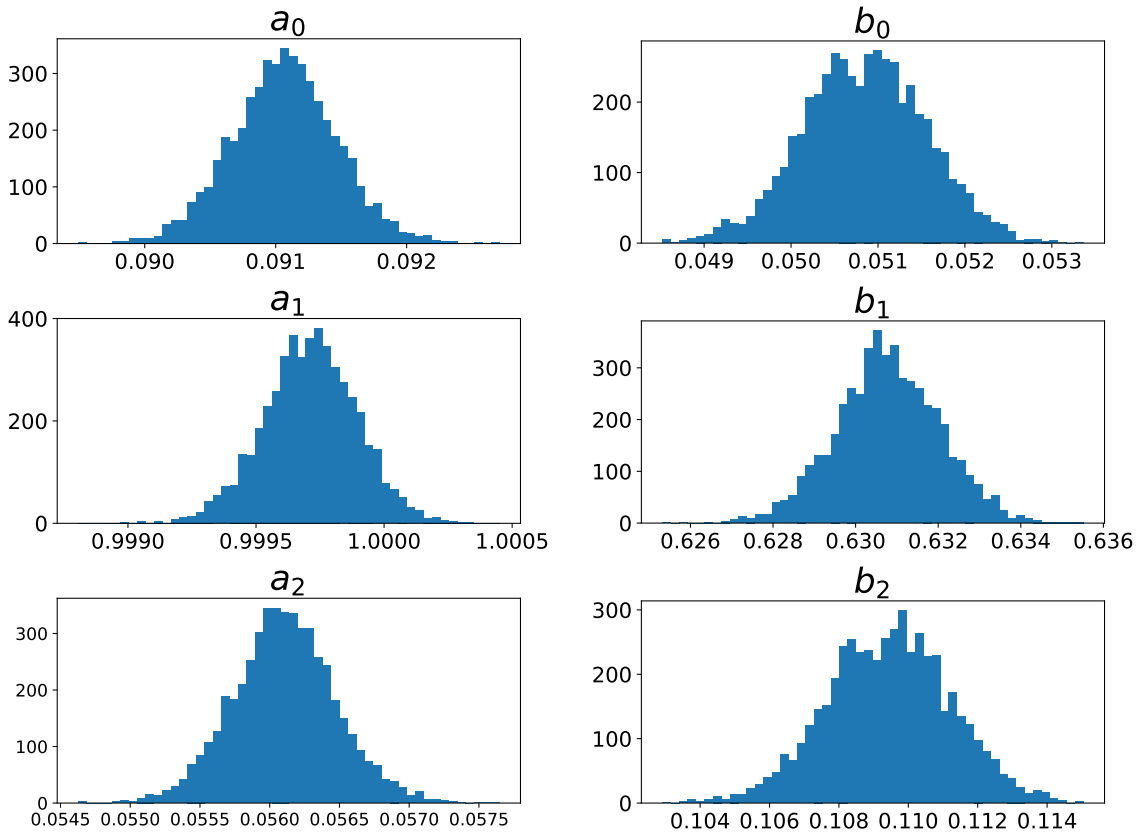
Table 3.1: Summary of the metamodels' inferences

Metamodel's parameters for half-circle			Metamodel's parameters for chevron		
Parameter	Mean	95% HPD interval	Parameter	Mean	95% HPD interval
a_0	0.091	[0.090, 0.092]	b_0	0.051	[0.050, 0.052]
a_1	0.9997	[0.9993, 1.0001]	b_1	0.631	[0.628, 0.633]
a_2	0.056	[0.055, 0.057]	b_2	0.109	[0.106, 0.113]

The parameters of the metamodel and their associated uncertainty can be estimated using a general probabilistic approach like Markov chain Monte Carlo, recursive Bayesian filters (e.g. Kalman and Particle filters) or least square methods [114–118]. Here, MCMC with the Metropolis-Hastings algorithm is used to draw samples of the metamodel parameter posteriors. Priors of the metamodel's parameters are defined based on available information from the original models, i.e., the priors for θ and β. The prior for the a_1 is selected identical to the prior for θ, and the prior for b_1 is selected identical to the prior for β. Also, the prior for the standard deviation can be obtained from standard deviation of these priors. The standard deviation of a uniform distribution between a and b is $\frac{1}{2\sqrt{3}}(b-a)$. This results in a uniform distribution bounded by zero and $\frac{1}{\sqrt{3}}$ for the half-circle model. For the chevron model, the prior of the metamodel standard deviation is a uniform distribution between zero and $\frac{\pi}{2\sqrt{3}}$ for the chevron model. The prior of the scale factor is selected as a normal distribution with the mean value of one and standard deviation of one.

The mean and the 95% credible interval for each parameter are summarized in Table 3.1. Based on these results, an estimate of the Bayes factor is obtained $\frac{0.091}{0.051} = 1.784$. Using the 95% HPD of these parameters we can say that the Bayes factor is most likely in the interval $[\frac{0.090}{0.052} = 1.731, \frac{0.092}{0.050} = 1.840]$. The "true" value of the Bayes factor was calculated as 1.833 using the Vegas Monte Carlo integration. This value is in the interval obtained by the surrogate-based method.

Figure 3.5 shows the marginal histograms of the MCMC chain. The histograms



(a) Half-circle (b) Chevron

Figure 3.5: Marginal histograms for the parameters of the metamodel

related to a_0 and b_0 can be used to obtain a histogram for the Bayes factor. This histogram, which is shown in Figure 3.6, represents a probabilistic description for the Bayes factor using the proposed method.

Concluding remarks

In this example, two one parameter models are compared using the Bayes factor. The Bayes factor is estimated with the proposed surrogate based method and compared with the Bayes factor calculated using classical Bayesian inference. A normal PDF times a scale factor is used as the metamodel of EDF. Since the area under a PDF is one, the scale factor approximates the model's evidence, eliminating the need for numerically integrating the evidence.

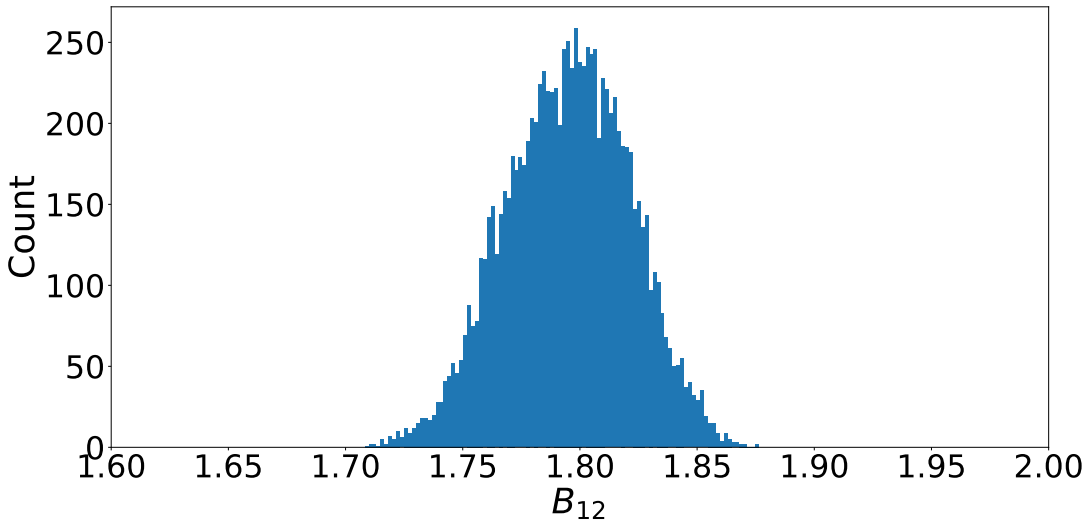


Figure 3.6: The Bayes factor histogram

The traditional Monte Carlo integration used 500 samples while the proposed technique used 11 and 14 evaluations of the models to satisfy the convergence criteria. A normal distribution was selected to approximate the integrand of the evidence function (EDF). A sampling strategy is discussed, and the stopping criteria was satisfied with 11 and 14 samples of EDF for half-circle and chevron model, respectively. The results of the proposed method show good consistency with those of the traditional Monte Carlo integration even though only a fraction of the model evaluations were required. The sampling method worked well on this example, but it is expected that this sampling strategy could have difficulties with peaked posteriors. Although a sampling strategy is not the main contribution of this work, in the Chapter 4 a new sampling strategy is introduced.

Chapter 4

Sampling Strategy

> "The theory of probabilities is at bottom nothing but common sense reduced to calculus."
>
> *Laplace (1820)*

4.1 Introduction

Depending on the problem and experience of the researcher, different sampling strategies may be chosen and applied to the proposed method. For instance, in each of the two examples that were solved in the previous chapters a different sampling strategy was used. In the first example (Page 21), samples are selected in a fixed step in the domain of the parameter. In that example, the samples are drawn from vicinity of the EDF peak and add meaningful information about that region to fit the metamodel. When the posterior and consequently EDF have a sharp peak, the value of EDF samples which are not in the vicinity of the peak is close to zero. A few number of those samples is needed to provide information for EDF tails. However, considering the computational effort required to evaluate models, adding additional samples in these regions does not improve the fitting process considerably. Having samples from a higher amplitude region of the EDF is more favorable.

In the second example (Page 29), initially some random samples were selected from a uniform distribution bounded by the parameter's lower bound and upper bound. These few samples, however, do not have enough information to help the metamodel

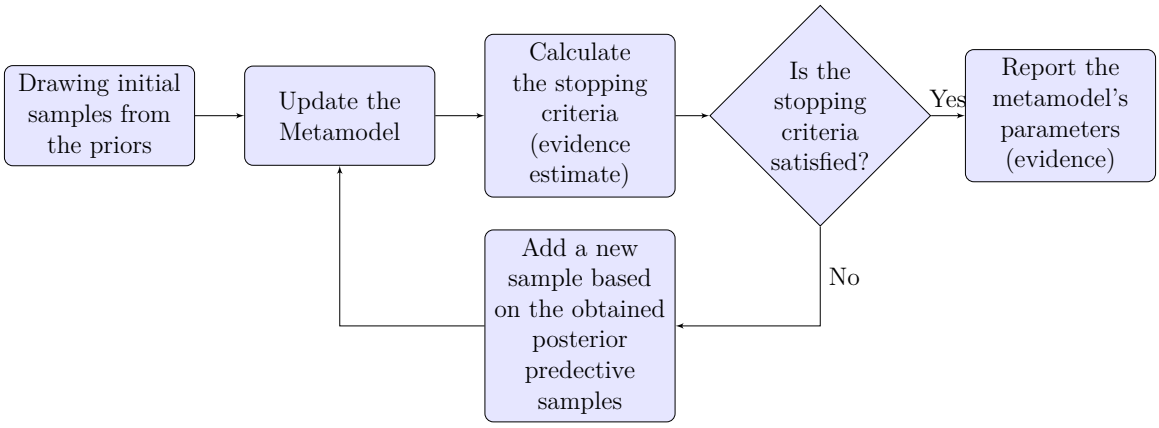
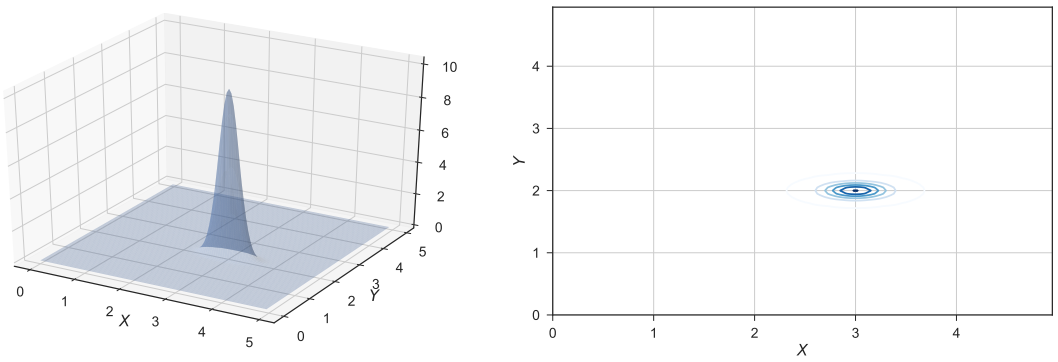


Figure 4.1: Sampling strategy flowchart

fit properly to the EDF, but can help the metamodel to define its tails. After the first fitting, new samples are drawn from the metamodel using a rejection sampling method. Since the metamodel shares the same tails with the true EDF, it is more likely that samples drawn in this way are located in the higher amplitude region of the EDF. In each iteration, the metamodel fits better to the EDF, and new samples are more likely to be drawn from the peak vicinity of the EDF.

This chapter suggests a more general method for sampling strategy. The sampling strategy for obtaining the model evidence can be summarized in the flow chart shown in Figure 4.1. First, initial samples are drawn from the EDF in domain of parameters. Then, an initial metamodel is updated using these samples. This metamodel is used to obtain the posterior prediaction for a set of guide points. The guide point which indicats the highest uncertainty in the posterior prediction is used to select the next adaptive sample. Parameters of the metamodel are updated based on the information provided by the new sample. The process is continued untill the stopping criteria is satidfied.

Two methods are proposed depending on how the guide points are defined. The sampling strategy is disscussed in more detail in the following sections.



(a) Three dimensional representation

(b) Contour plot

Figure 4.2: The test function

4.2 Posterior predictive based sampling strategy

Here, an example is used to explain the sampling strategy. In this example, a test function is defined by the equation $f(x, y) = [\phi(x-3, 0.5)\phi(y-2, 0.2)]^5$, where $\phi(a, b)$ represents a normal distribution with mean a and standard deviation b. The goal of this example is to fit another function, $g(x, y) = \rho\phi(x - \mu_x, \sigma_x)\phi(y - \mu_y, \sigma_y)$, to this function, where ρ is the scale factor. The test function represents a "true" model, e.g. EDF, and $g(x)$ plays the role of the metamodel. The domain of the parameters, or the design space, is defined from zero to five in both x and y directions. Figure 4.2 shows a graphical representation of $f(x, y)$ with a sharp peak at $x = 3$ and $y = 2$. The sampling strategy steps to fit this metamodel are explained below.

First step: Drawing random samples

Some initial samples are drawn from the true model in a random fashion. To do that, a uniform distribution is defined for x and y directions. Twenty samples are selected from each of the distributions. Each sample is considered as an input for the test function and the output is considered as data obtained from the true model. To simulate the measurement error, a small random number from a normal distribution is added to the output. These initial samples are shown in Figure 4.3. Since the true

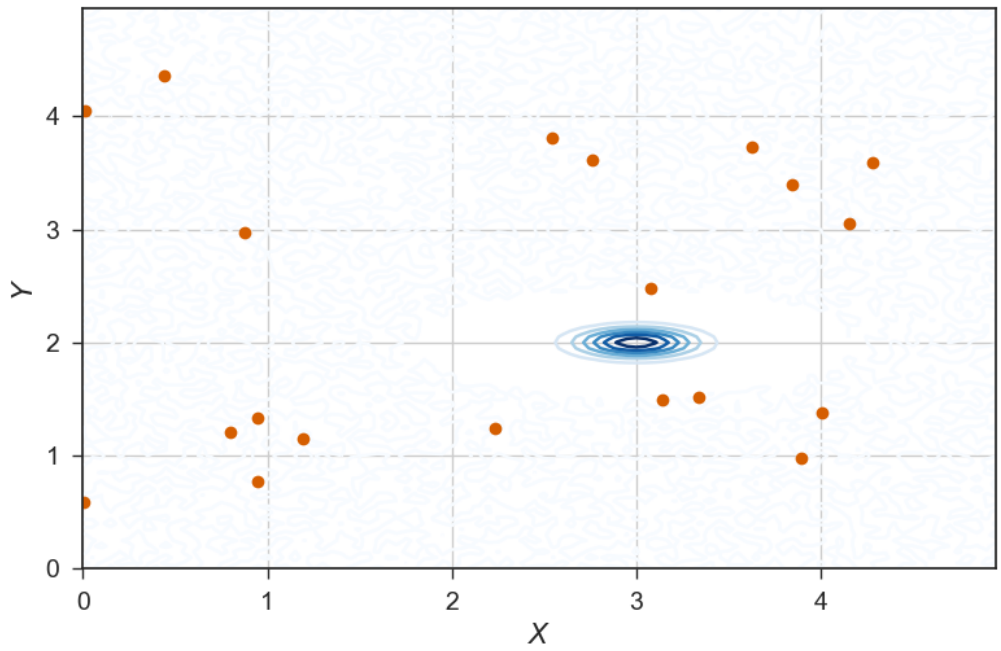


Figure 4.3: Initial samples which are selected in a random fashion

function has a sharp peak, none of these samples are located in the peak vicinity. In other words, these samples are not usually able to provide considerable information about the peak's area. However, they have information about other regions of the test function and we can use them to fit the metamodel.

Second step: Metamodel updating using the initial samples

In this step, it is intended to fit the metamodel to the test function. To do that, a Bayesian model is defined. There are six parameters to update: μ_x, σ_x, μ_y, σ_y, ρ, and standard deviation of the likelihood, σ_{lik}.

For each of these parameters, a prior should be defined. For the mean values, a uniform distribution from zero to five is selected as the prior. For the standard deviations, a gamma distribution with a mean value of 1.5 is selected. For the scale factor, a normal distribution with a mean value of 1.0 and a standard deviation of 0.5 is selected. A potential could be considered to define zero as the lower bound, although it was not used in this analysis.

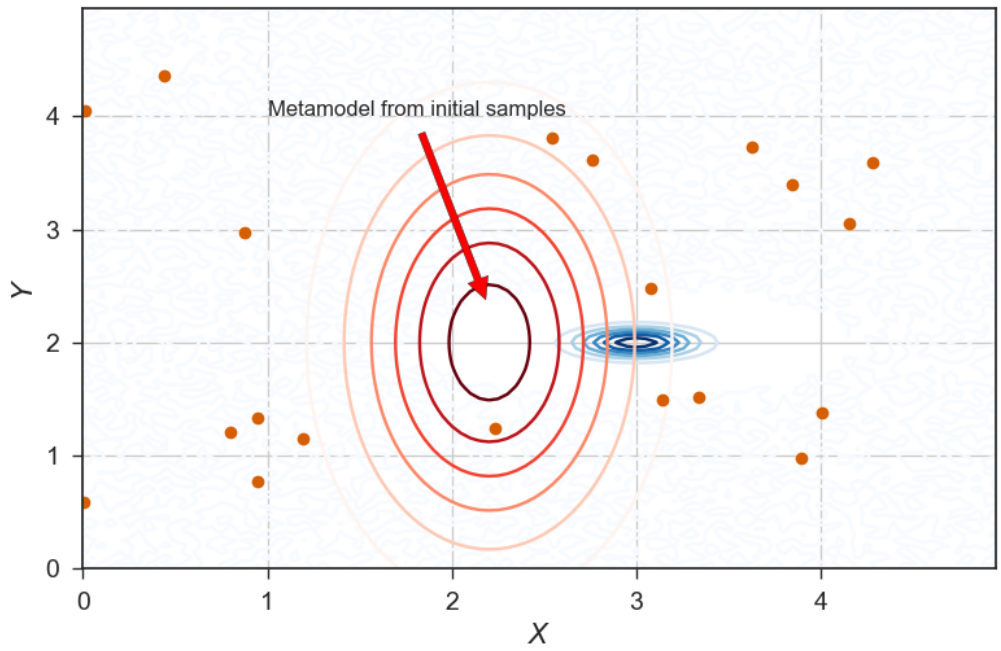


Figure 4.4: Contour of the obtained metamodel from initial samples on the true model contour.

Selecting the initial point of the MCMC chain is an important part of the analysis. Poor initial values may increase burn-in time and delay the MCMC chain convergence [119, 120]. To find good starting values, we tried to find the values which maximize the posterior. In this work, the metaheuristics optimization method of differential evolution (DE) is used to find the maximizer values. [121, 122].

The updated metamodel based on the initial points is often not a good representation of the true model as shown in Figure 4.4. The metamodel updated with the initial samples is shown in red and the true model in blue. However, this metamodel could be used to select the location where samples of the "true" model might yield the most information.

Third step: Best candidate for the next sample

Predictive posterior distribution

Recalling Equation 1.3, $P(\theta)$ is the initial belief about the parameters of the model. Taking available samples into account, D, the initial belief about parameters would change and a better representation of the uncertainty is obtained.

$$P(\theta|D) = \frac{P(\theta)P(D|\theta)}{P(D)} \tag{1.3}$$

Furthermore, this updated belief can be used to calculate the probability of observing new data, $P(D_{new}|D)$, using Equation 4.1 [123]:

$$P(D_{new}|D) = \int_{\theta} P(D_{new}|\theta, D)P(\theta|D)d\theta = \int_{\theta} P(D_{new}|\theta)P(\theta|D)d\theta \tag{4.1}$$

MCMC methods are used to sample $P(D_{new}|D)$. Obtained samples are called the posterior predictive samples.

Guide Points Set

Posterior predictive distribution can be used to model the likely values of new data at specific locations in the design space [110]. In this work, I consider a grid of preset locations in the design space, Guide Points Set (GPS), and treat them as missing data. The predictive posterior samples for these points are obtained. Less information about these points results in more uncertainty for their corresponding posterior predictive samples. As a result, I selected the next sample in the vicinity of a guide point with the largest standard deviation in its posterior predictive samples.

In our example, a mesh of 30 by 30 is selected for the GPS as shown in Figure 4.6. Each time where the Bayesian model is run to fit the metamodel, posterior predictive samples are obtained. Standard deviation of samples at each point is calculated, and the next sample from the true model is selected in the vicinity of the point with the largest standard deviation. By adding a new sample, the metamodel is updated,

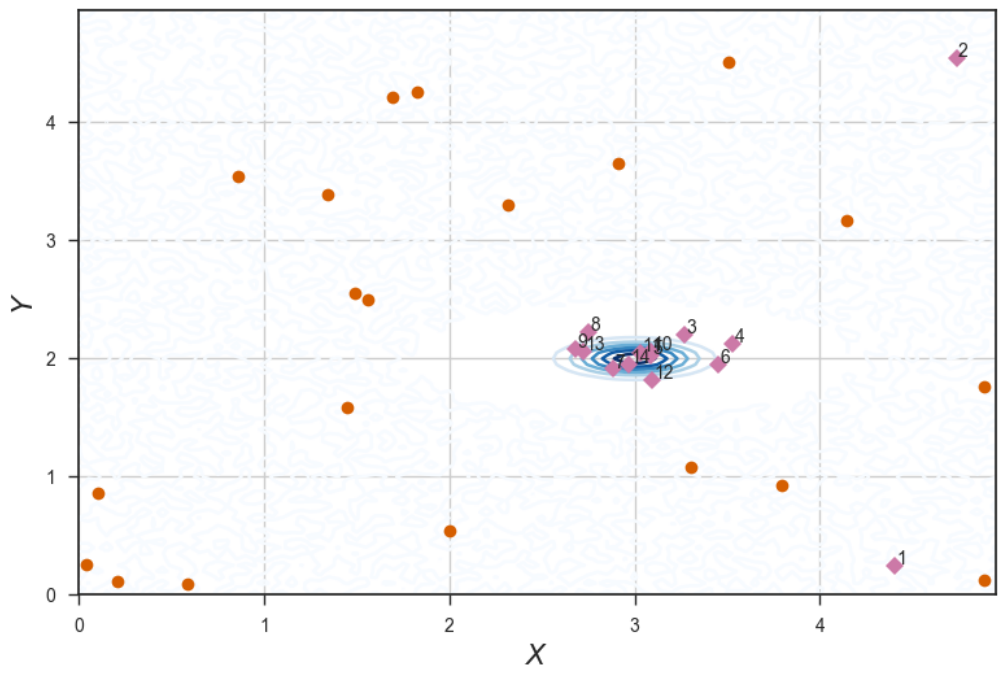


Figure 4.5: New samples are added using the GPS.

and new posterior predictive samples are obtained for GPS. Repeating this process, enough samples are added to satisfy the stopping criteria. Figure 4.5 shows the added samples to initial samples. The number next to the added samples indicates the order of added samples. Figure 4.7 is related to the guide point indicated by the arrow in Figure 4.6. The box plots correspond to mean zero (samples value minus their average) posterior predictive samples in each iteration. It shows how the dispersion of these samples are changed by adding new samples. The general trend shows that when a new sample is selected close to the indicated guide point, it decreases the desperation on posterior predictive samples by providing more information about that region.

Last step: Stopping criteria

Adding samples should be continued until the stopping criteria is satisfied. In this example, the stopping criteria is defined on the scale factor. When the sam-

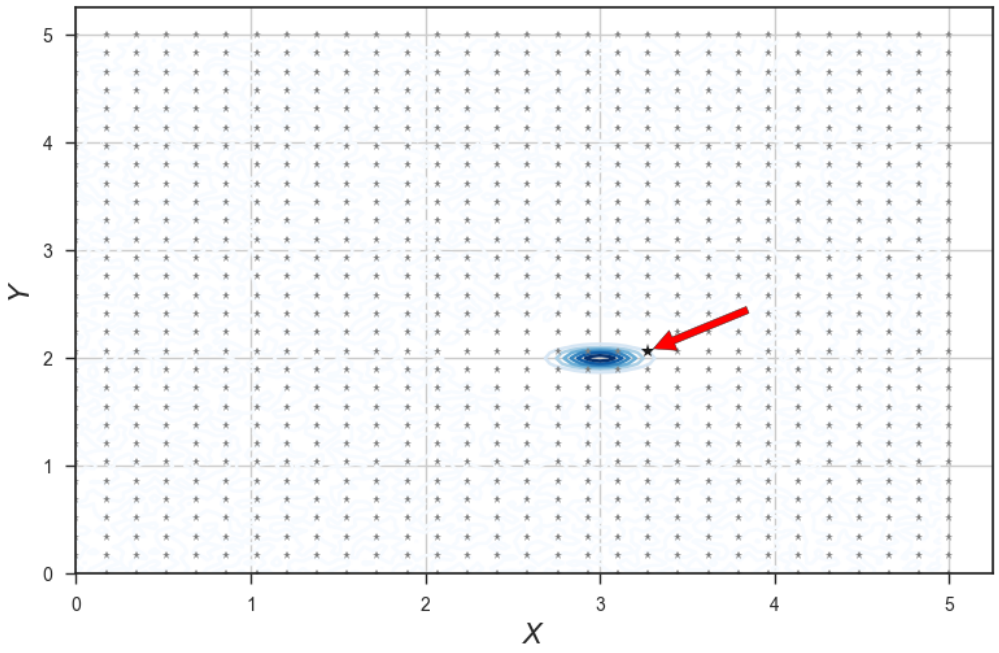


Figure 4.6: GPS mesh on the design space

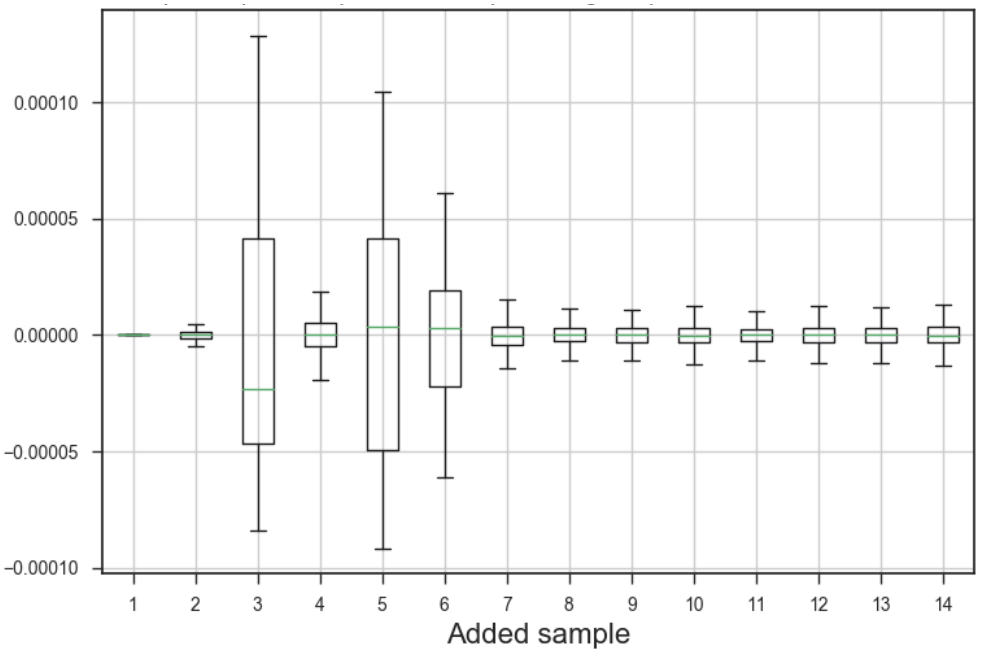


Figure 4.7: Box plots for the posterior predictive samples of one of the guide points.

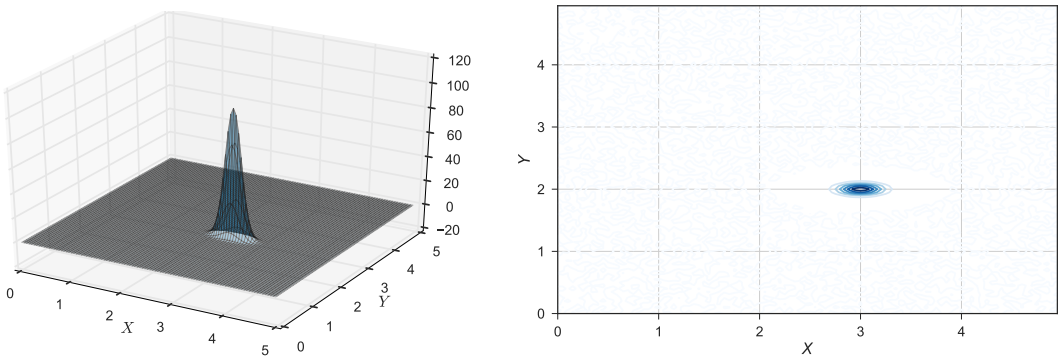
pling strategy is used for model selection problems, the stopping criteria is defined on the model evidence. One of the problems with this sampling strategy is that, sometimes, adding several subsequent samples does not add enough information; then, the selected parameter for the stopping criteria does not change considerably in two subsequent iterations. So, the stopping criteria is satisfied, while the metamodel is not fitted properly. In this case, we can reduce the stopping threshold and also stop the process when the stopping criteria reaches the stopping threshold for several times in a row. In the example, the stopping criteria is defined as shown in Equation 4.2, and it should be satisfied five times in a row to stop the process. In the Equation 4.2, ρ_{mean} is the mean value for the scale factor which is obtained from its corresponding MCMC samples.

$$\frac{\rho_{mean}(i+1) - \rho_{mean}(i)}{\rho_{mean}(i+1)} <= 0.005 \tag{4.2}$$

The stopping criteria works well for the mentioned example. The whole process was repeated five times from the same initial samples, and each time after adding approximately 15 samples, the stopping criteria was satisfied. The metamodel checked and it was fitted properly to the true model. More advanced stopping criteria may be considered in different problems. For example, I noticed in the case that sampling strategy works well, last added samples are more localized than initial added samples (for example see Figure 4.6). So, defining a stopping criteria for the distance between two subsequent samples could be considered as an alternative to the proposed stopping criteria.

To investigate the robustness of the proposed sampling strategy, another test function is examined with a sharper peak as described by Equation 4.3 and Figure 4.8. The stopping criteria is set to 0.001.

$$f(x, y) = [\phi(x - 3, 0.5)\phi(y - 2, 0.2)]^{10} \tag{4.3}$$



(a) Three dimensional plot

(b) Contour plot

Figure 4.8: Test function with a sharper peak

To fit the model, the same initial samples as the previous example are selected (Figure 4.3). Iterations are stopped after obtaining a few samples, but the metamodel is not fitted properly to the test function in the first few fitting cycles. Different number of cycles might be required to satisfy the stopping criteria when the process is run more than one time due to the random nature of the adaptive samples. A total of 15 to 30 cycles were required for this particular problem. The metamodel fit the true model appropriately in all attempts . The contour of the metamodel, in addition to added samples, is shown in Figure 4.9 . In the next section, a structural model selection example is presented to show how the proposed sampling stratyegy is used to sample EDFs.

4.3 Example: A structural example

In this example, two beam models are used to describe the data obtained from end displacement of a beam. Models have two parameters, and it is shown that the proposed method can estimate the Bayes factor with less computational effort.

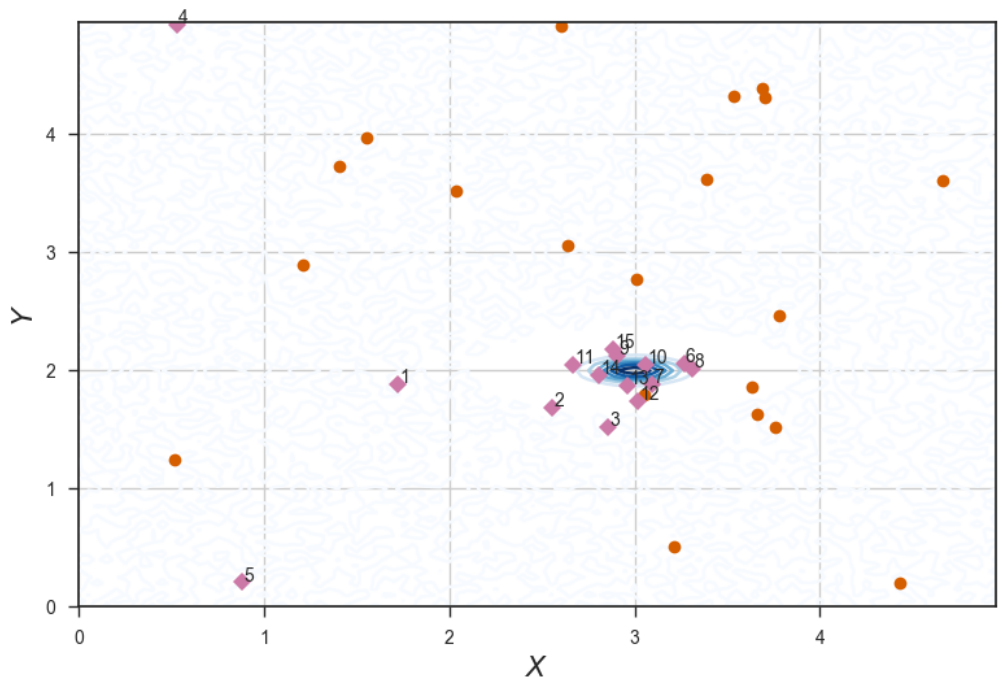


Figure 4.9: New samples are added using the GPS in the test function with a sharper peak.

Problem Definition

Figure 4.10 shows the model representing the "true" structure. Data is obtained by measuring the end displacement of the beam when a point load, P, or moment, M, is exerted to the end of the beam. The parameter θ is the ratio of the cantilever part of the beam to the whole length of the beam. The "true" model has a value of $\theta = 0.5$. A normally distributed number with a standard deviation of 0.005 m is added to the end displacement as the measurement's noise. The parameter of the interest is θ, and two models are proposed. The first model is the same as the "true" structure. The second model considers constraining the rotational DOF of the internal support. Table 4.1 shows different loadings and their corresponding end displacement of the beam.

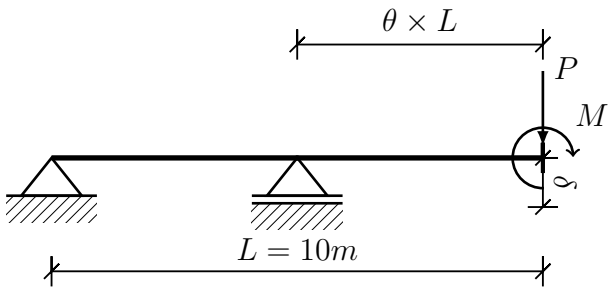


Figure 4.10: The first model with a simple support within the length of the beam

Table 4.1: Data used in the example

P (N)	M (N.m)	Displacement (m)
20000	0	0.0855
20000	0	0.0817
0	50000	0.0642
0	50000	0.0508

Monte Carlo Estimate

Bayes factor is the ratio on the Bayesian evidence of the two models. Here, normal distribution is assumed for the likelihood, so standard deviation of the likelihood, σ, is considered as a free parameter.

To calculate the evidence for each model Monte Carlo estimates of each integral using the Vegas algorithm is obtained [104]. The algorithm is stopped after 3 iterations, when the error is approximatly 10%. A total of 3000 model evaluations are used in this process. The Bayes factor is calculated as the ratio of the evidences, $B12 = \frac{95166}{52777} = 1.803$, which shows a preference for model 1 as expected. Table 4.2 summarizes the obtained results by implementation of the Vegas algorithm.

Proposed Surrogate based Method

The evidence in our example contains two parameters (θ, σ). Therefore, a bivariate distribution is used to model the EDF. Based on our experience, we know that the distribution for σ is usually skewed. So we selected a bivariate distribution with a normal distribution for θ and an inverse Gaussian distribution for σ domain.

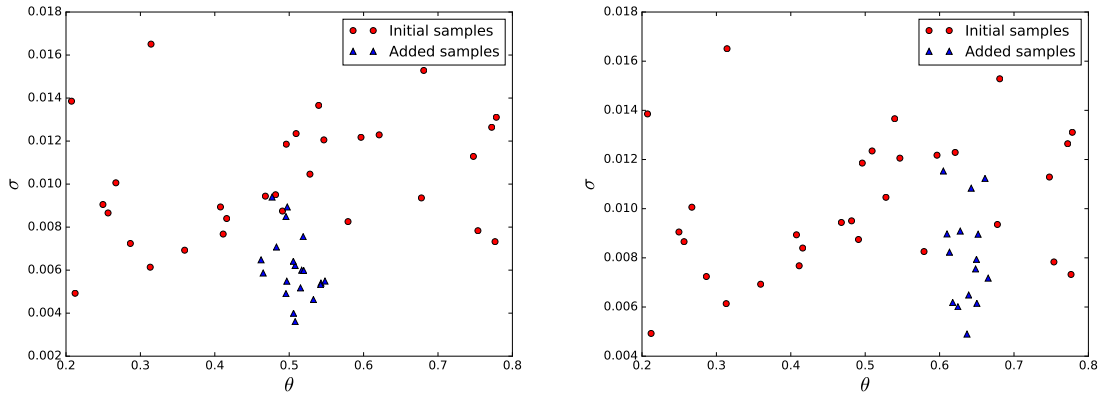
Table 4.2: Monte Carlo integration

Iteration	Obtained value	Error	P-value	Total number of samples
Model 1				
1	89798	10479	1.00	1000
2	94862	2506	0.25	2000
3	95166	1023	0.13	3000
Model 2				
1	40007	9028	1.00	1000
2	52346	2280	0.16	2000
3	52777	587	0.36	3000

Seven parameters needed to be updated including: mean and standard deviation for normal distribution; and mean, scale and shape parameters for the inverse Gaussian distribution; the scale factor to fit the metamodel; and standard deviation of the likelihood of a Bayesian model used to fit the metamodel. Samples of the EDF, requiring evaluations of the structural model, are drawn using the following strategy. First, 30 samples are randomly selected from priors of θ and σ. Then, a mesh of 900 guide points, GPS, are constructed in the θ and σ domain, and posterior predictive samples are obtained for each of these points. The next sample of the EDF is obtained from the cell with the highest standard deviation of the predictive samples. The process is stopped when the estimated evidence does not change significantly between steps. 50 and 45 samples are obtained for models 1 and 2, respectively. Figure 4.11 shows the samples drawn for each model. Figure 4.12 shows the fitted metamodels on EDF of model 1 and 2. Finally, MCMC chains are obtained for all parameters of the metamodel, and the Bayes factor is calculated from these samples as indicated in a histogram shown in Figure 4.13.

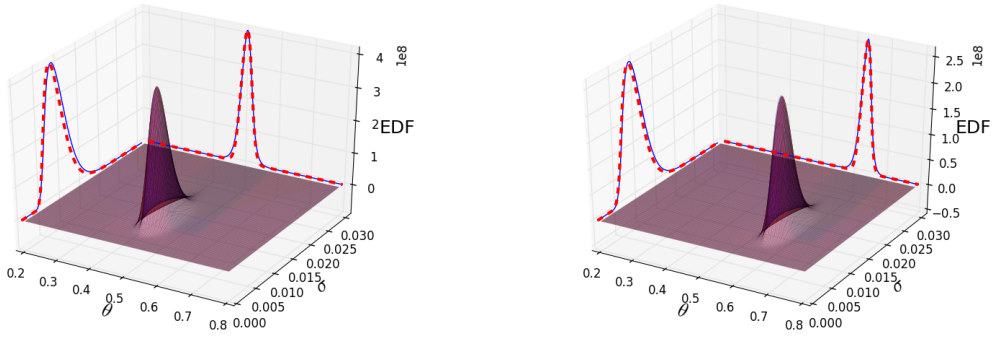
Concluding Remarks

The Bayes factor to compare two structural models is calculated using two different methods: 1)Using a traditional MCMC algorithm (Vegas) using 3000 evaluations of



(a) Model 1 (b) Model 2

Figure 4.11: Sampling for both models



(a) Model 1 (b) Model 2

Figure 4.12: Fitted Metamodel for model 1 EDF (red surface and dashed red lines are related to the metamodel

the structural model, and 2) Using a proposed technique that uses a metamodel of the EDF and only requires 45 and 50 evaluations of the structural model 1 and 2, respectively. Results of the proposed technique show that there is a 95% probability that the Bayes factor is between 1.792 and 1.816. These results include the value of 1.803 obtained with the Vegas algorithm. The proposed methodology has the potential to reduce the computational cost of model selection of computationally expensive models as it only uses a fraction of the samples which usually are used when traditional methods are implemented.

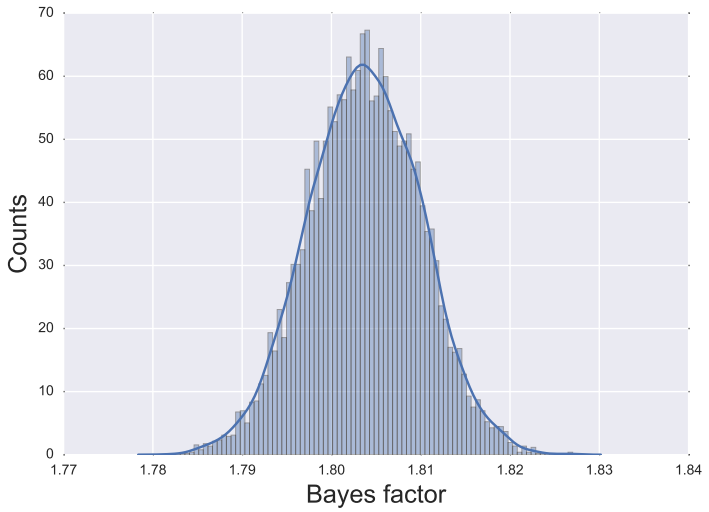


Figure 4.13: Histogram for Bayes factor

4.4 Posterior predictive sampling technique with evolutionary GPS

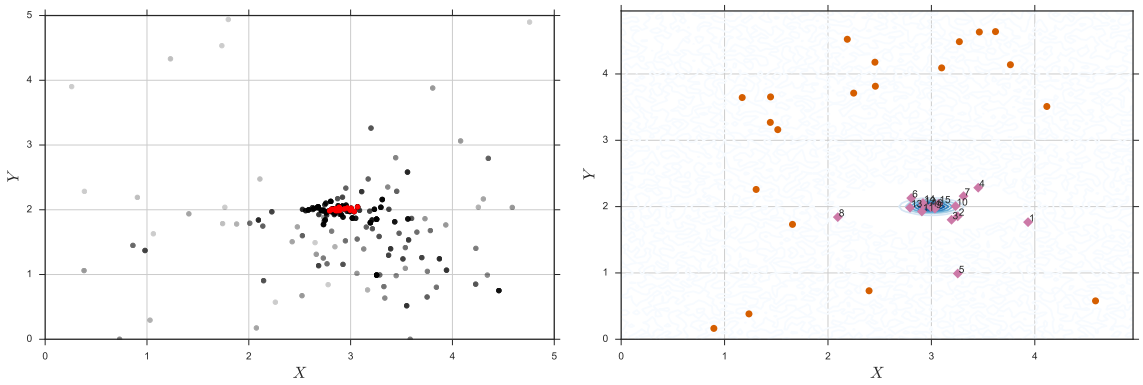
One of the challenges in implementation of the described method is the size of the GPS. In the example that is solved above, 900 points are used for GPS. It means that for each of those points, posterior predictive samples should be calculated. If we want to have a similar mesh of GPS in higher dimension problems, the size would increase exponentially, e.g., for a five dimensional problem, $30^5 = 24.3e6$ guide points are needed. This makes the implementation of the methodology very expensive, such that in some examples, it demands even more computational effort than the direct implementation of the true model. Selecting the GPS in a mesh form is the simplest way to implementation of the proposed sampling strategy. Alternatively, one can define a strategy for selecting the GPS.

The idea is that to let the guide points decide their location on their own by using the information they provide in each run. After adding the initial samples, some points are selected from the domain of parameters in the random fashion. The number of the points are selected in a way that obtaining posterior predictive samples for them does not add a considerable computational effort to the problem. In each

run a new sample is added to the initial samples. This sample would be added at the guide point which standard deviation of its related posterior predictive samples is the maximum. The standard deviation of the posterior predictive samples is considered as an uncertainty indicator. Then, in each run, a portion of guide points, which their corresponding posterior predictive samples have smaller standard deviation, is disregarded. It is assumed that the reason they have less standard deviation is that there is less uncertainty about those points. Instead of them, new guide points are added in the vicinity of the location with more uncertainty.

In the second example, which was solved in the previous section, 900 guide points had been used. Here the same example is solved using only 20 guide points. Similar to the previous example, 20 initial samples are drawn from the parameters' domain. In each run, 10 of the guide points, which their related posterior predictive samples have smaller standard deviation, are discarded and 10 new guide points are added instead of them. New guide points are drawn from a normal distribution which is defined based on the mean and standard deviation of the remaining guide points. After adding 16 samples to the initial samples, the stopping criteria is satisfied. Figure 4.14 shows the sampling strategy implementation. Figure 5.3a shows the evolution of the guide points by a change in the darkness of points. The brighter points related to initial GPSes and darker sets show the final GPSes. The last GPS is indicated by the red color. Figure 4.14 shows the added samples order.

The evolutionary sampling strategy can zoom on the location where more information is needed for the sampling. This a good feature when there is a sharp peak in the fitting problem. Here, I selected the new guide points based on the information from the previous run. In some problems, probably when there are several uncertain locations in the domain of parameters, one can divide new guide points into two subsets. The first subset could be selected based on the previous guide points, while the second subset could be selected based on the parameters' domain (the same as the



(a) GPS Evolution (b) Added samples

Figure 4.14: Evolutionary sampling strategy

first GPS). By this strategy, all of the guide points do not zoom in only a specific region, such as shown in Figure 5.3a by the red color. The examples are solved in the next chapter use the evoulutionary sampling strategy to select the adaptive samples.

Chapter 5

Metamodel Generalization

> "Laplace [used] ... Bayes' theorem ... to help him decide which astronomical problems to work on.... in which problems is the discrepancy between prediction and observation large enough to give a high probability that there is something new to be found?"
>
> *E. T. Jaynes (1986)*

5.1 Introduction

One of the challenges with the introduced method in this dissertation is how to select the appropriate surrogate model for the EDF. In the previous chapters, it was shown that in some examples, EDF is a skewed function (e.g. when the standard deviation of the likelihood is taken as a free parameter). For these cases, a skewed metamodel can be selected as a surrogate model; however, selecting a good candidate depends on the experience of the analyst. This chapter suggests using Gaussian mixture model (GMM) as the surrogate model. The nature of the proposed methodology allows for the use of mixture of Gaussian distributions without adding further complexity for implementation.

A distinction needs to be made between a mixture of distribution functions and a random variable whose value is obtained by adding up the values of random variables from different distributions. Here, the focus is on the former, which is simply adding the Gaussian probability densities by different weight of scale factors. Adding weighted random variables from different Gaussian distributions generates a new Gaussian random variable, while adding their density functions generates a function which may not be Gaussian anymore. GMM are intensively used to model the arbitrarily shaped densities [109].

5.2 Mixture of Gaussian distributions

Bayes factor is defined as the ratio of two model's Bayesian evidences, and Bayesian evidence is defined as:

$$p(R|M) = \int p(R\theta|M)d\theta = \int p(R|\theta M)p(\theta|M)d\theta \tag{5.1}$$

where R is the observed data for model M. The likelihood, $p(R|\theta M)$, is a function of the data given the parameters, θ. The GMM is used to approximate the integrand in equation 5.1.

$$g = \sum_{i=1}^{n} \rho_i \phi_i \tag{5.2}$$

In the above equation ϕ_i is the i th normal density function in the GMM. Replacing the EDF in Equation 5.1 by g in Equation 5.2, the model evidence, B_M, is obtained as:

$$B_M = \int \sum_{i=1}^{n} \rho_i \phi_i d\theta \tag{5.3}$$

This integral is solved straight forward since the integral of a normal density function is its corresponding CDF. For example, consider a case in which EDF is the function of two independent variables, θ_1 and θ_2, respectively, and using n numbers of Gaussians,

the metamodel is defined as follows:

$$g = \sum_{i=1}^{n} \rho_i \phi(\theta_1 - \mu_{\theta 1i}, \sigma_{\theta 1i}) \phi(\theta_2 - \mu_{\theta 2i}, \sigma_{\theta 2i}) \tag{5.4}$$

In the Equation 5.4, ρ_i, $\mu_{\theta 1i}$, $\sigma_{\theta 1i}$, $\mu_{\theta 2i}$, and $\sigma_{\theta 2i}$, for $i = 1, ..., n$ are the metamodel's parameters. Then the evidence is calculated by measuring the volume under this metamodel as follows:

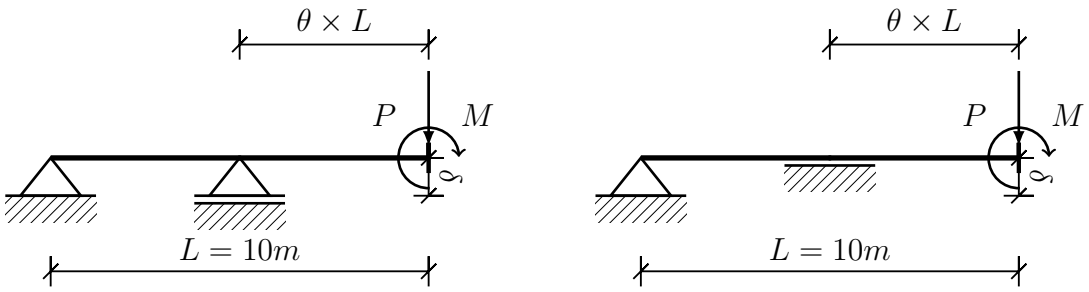
$$\begin{aligned} \int_{\theta_{1d}}^{\theta_{1u}} \int_{\theta_{2d}}^{\theta_{2u}} \sum_{i=1}^{n} \rho_i \phi(\theta_1 - \mu_{\theta 1i}, \sigma_{\theta 1i}) \phi(\theta_2 - \mu_{\theta 2i}, \sigma_{\theta 2i}) d\theta_1 d\theta_2 = \\ \sum_{i=1}^{n} \rho_i \Phi(\mu_{\theta 1i}, \sigma_{\theta 1i})|_{\theta_{1d}}^{\theta_{1u}} \Phi(\mu_{\theta 2i}, \sigma_{\theta 2i})_{\theta_{2d}}^{\theta_{2u}} \end{aligned} \tag{5.5}$$

Here Φ is normal cumulative distribution function, and θ_{1id}, and θ_{1iu} are limits of the integral in θ_{1i} domain. In the same way θ_{2id} and θ_{2iu} are integral limits in the θ_{2i} domain. The integral would be simpler if the priors are defined on infinite domain i.e. $\theta_{1id} = \theta_{2id} = -\infty$, and $\theta_{1iu} = \theta_{2iu} = \infty$. In this case, the evidence would be obtained simply from the summation of scale factors, or $\sum_{i=1}^{n} \rho_i$ based on the second axiom of the probability.

5.3 Example: Using GMM as a metamodel

Example definition

Two Euler-–Bernoulli beams are considered to explain how the methodology works. Each beam has two supports and there is a parameter that indicates the location of the middle support from the free end of the beams. Beams are 10 m long and have a moment of inertia of $1e-4$ m^4 and a Young's modulus of 200 GPa . End displacements of the beam with simple support at the middle of the span (i.e. $\theta = 0.5$) are considered as the data. A random noise with standard deviation of five millimeters, is added to the displacements for simulating the measurement error. Loading scenarios and displacements before and after adding noise are summarized in Table 5.1



(a) First model (b) Second model

Figure 5.1: Two models

Table 5.1: Loading and displacements

P	M	Displacement before adding noise	Displacement after adding noise (δ)
10000 N	0	0.0416 m	0.0376 m
10000 N	0	0.0416 m	0.0433 m
0	20000 $N.m$	0.0208 m	0.0207 m
0	20000 $N.m$	0.0208 m	0.0240 m

The end displacement of the beams shown in Figure 5.1 are described mathematically as follows:

$$Model\ 1 : \Delta_1(\theta, p, M) = \frac{l^2}{EI}(Pl\frac{\theta^2}{3} + M(\frac{\theta^2}{6} + \frac{\theta}{3})) \tag{5.6a}$$

$$Model\ 2 : \Delta_2(\theta, p, M) = \frac{l^2}{EI}(Pl\frac{\theta^3}{3} + M\frac{\theta^2}{2}) \tag{5.6b}$$

Here θ is the ratio of the length of the beam after the support to the total length of the beam, as indicated in Figure 5.1.

Bayesian model selection

To make a Bayesian model for these beams, we needed to assume priors for θ and the standard deviation of the likelihood. The value of θ is assumed a value between 0.2 and 0.8, and it is unlikely to see the θ out of this interval based on our experience. Also it is assumed that the error of measurement is in order of millimeter. So the

prior for the likelihood standard deviation is assumed a positive value distribution with mean value in order of millimeter. These priors are as follows:

$$\theta \sim Uniform(0.2, 0.8) = 1.667 \tag{5.7a}$$

$$\sigma \sim Gamma(2, 0.002) = 2500 \times \sigma e^{-500\sigma} \tag{5.7b}$$

Here 0.2 and 0.8 are lower bound and upper bound of the Uniform distribution, respectively, and 2 and 0.002 are shape and scale parameters of the Gamma distribution with a mean value of 4 mm. Using these priors, the evidence (B_j) for each model ($j = 1, 2$) is obtained from the following equations:

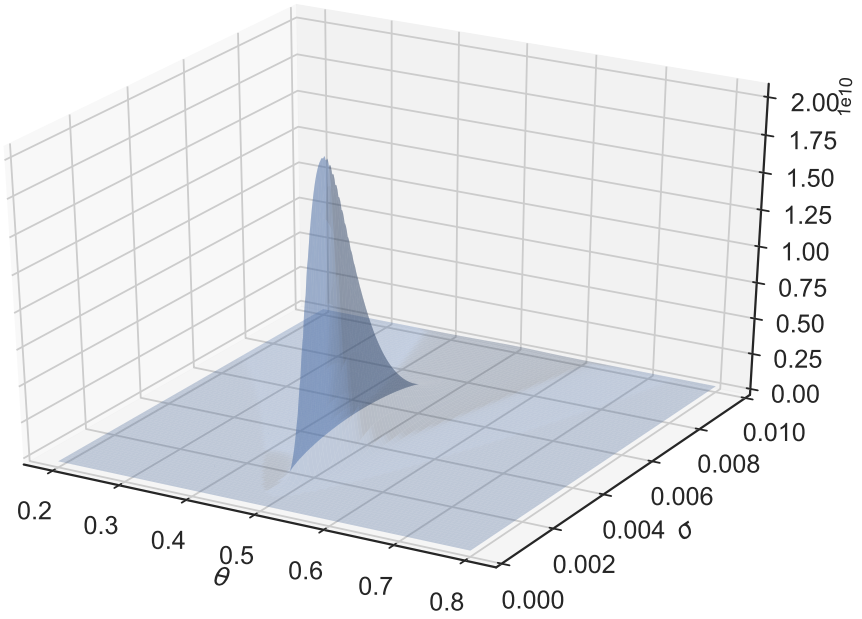
$$B_j = \int\int \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-\sum_1^4 (\Delta_j(\theta, p_i, M_i) - \delta_i)^2}{2\sigma^2}} \times 1.667e6 \times \sigma e^{-1000\sigma} d\theta d\sigma \tag{5.8}$$

Numerical calculation of the Bayes factor

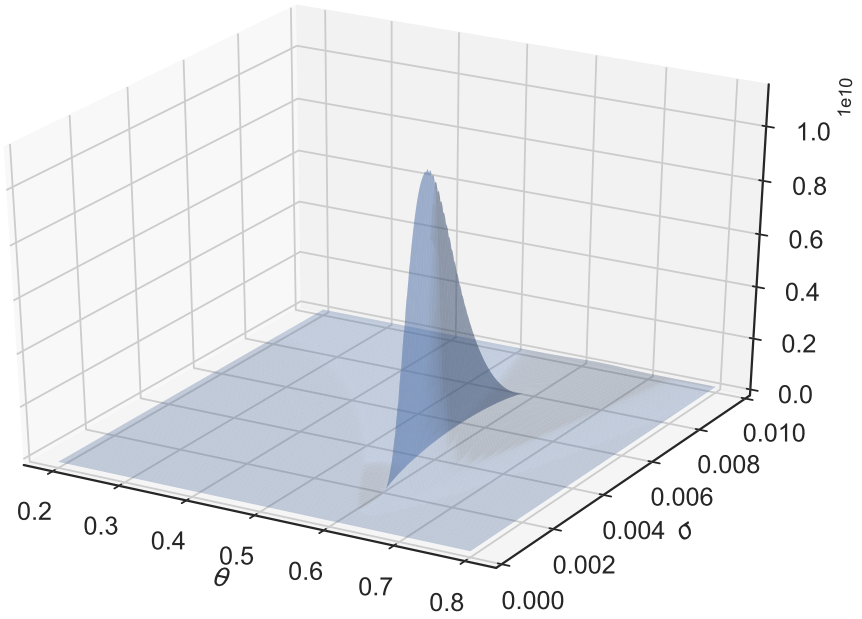
Figure 5.2 shows the EDF for both models, the volume under these surfaces is the models' evidence. Volumes can be calculated directly using numerical methods. Here the Vegas algorithm, which is one of the Monte Carlo integration methods, is used to evaluate this volume [104]. To reach to the convergence, 10 iterations with maximum number of 1000 integrand evaluation per iteration are done. For the first model and the second model, 95% of confidence intervals are obtained as $[1611089.33, 1627532.44]$, and $[891197.28, 900478.42]$, respectively. Bayes factor can be obtained using equation 5.8 as ratio of B_1 to B_2, $[\frac{1611089.33}{900478.42} = 1.789, \frac{1627532.44}{891197.28} = 1.826]$ which suggests more evidence in favor of model 1.

Metamodeling using GMM

To implement GMM, two different metamodels are considered. The first GMM contains only one Gaussian function, the second GMM has three Gaussian functions.

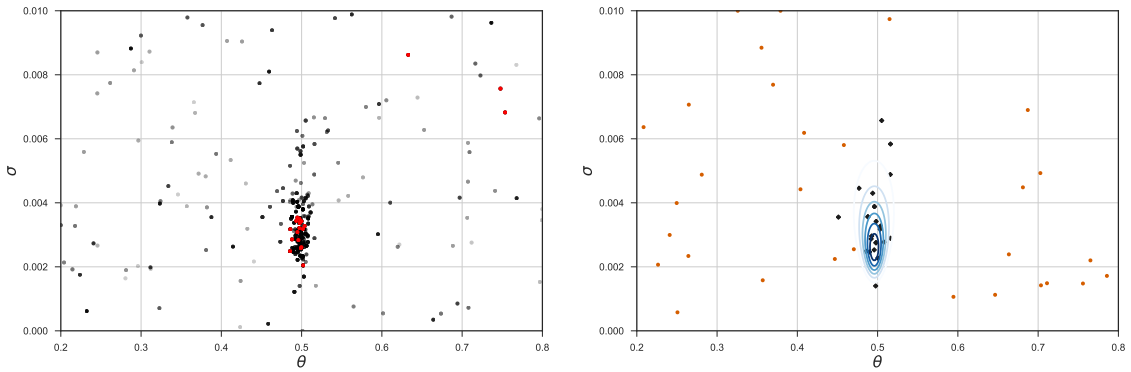


(a) Model 1



(b) Model 2

Figure 5.2: EDF for each model



(a) GPS evolution for Model 1

(b) Obtained samples

Figure 5.3: Sampling

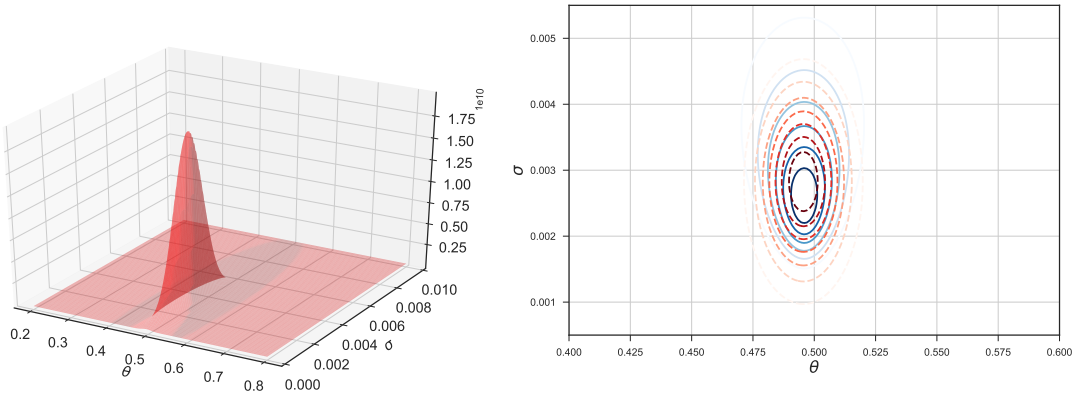
One Gaussian function:

GMM model with only one Gaussian function is shown in Equation 5.9:

$$\rho\phi(\mu_\theta, \sigma_\theta)\phi(\mu_\sigma, \sigma_\sigma) \tag{5.9}$$

A Bayesian model is defined to obtain the metamodel's parameters and standard deviation of the likelihood, σ_{lik}. The sampling strategy which is explained in the previous chapter is used to select samples from the true model. 30 initial samples are selected, and then subsequent samples are selected based on predictive posterior distribution at GPS points. 20 points are selected for GPS and 10 of them are replaced by new points in each iteration based on the explained sampling strategy in the previous chapter. For the first model, evolution of GPS is shown in Figure 5.3a by increasing the darkness in the points' color, and the last set of points is shown in the red color. 30 initial samples and subsequent 24 samples are shown by brown circles and black diamond shape markers in Figure 5.3b on the contour plot of the first model EDF. Similarly, as it shown in Figure 5.5, for the second model a total of 52 samples were needed to satisfy the stopping criteria.

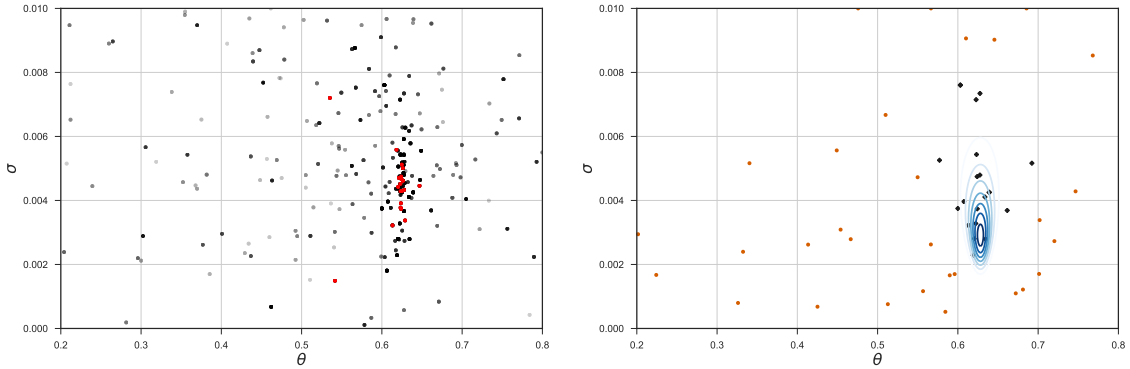
Figure 5.4a shows the obtained metamodel for model 1, and Figure 5.4b shows the metamodel contour (red dash line) which is plotted on the true model contour (blue solid line). Since only one Gaussian was selected, no skewness is observed in the



(a) Three dimensional plot of the metamodel

(b) Contour plot of the metamodel (dash line) on contour plot of the true model

Figure 5.4: One Gaussian Metamodel for model 1



(a) GPS evolution for Model 2

(b) Obtained samples for Model 2

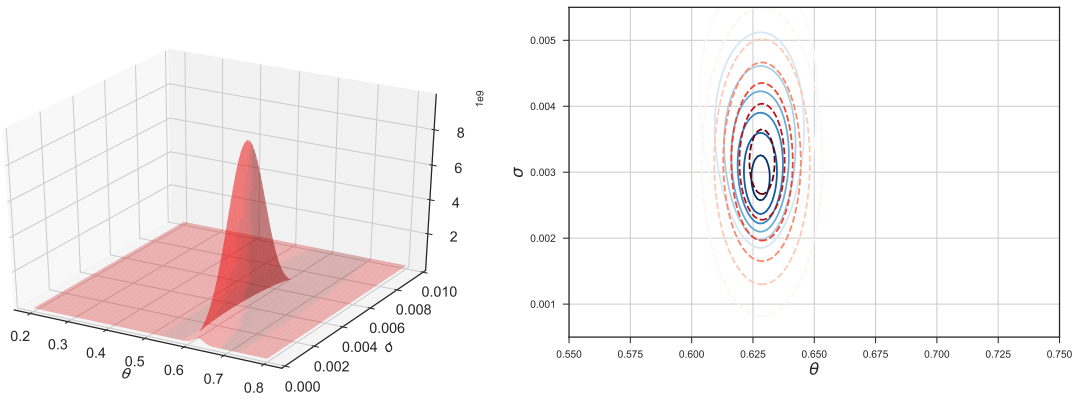
Figure 5.5: Sampling for model 2

results, while the true EDF has skewness. Figure 5.6 shows the metamodel obtained for model 2, and it shows related contour plot on the true model. After performing the MCMC for each of these models, the related chain for Bayesian evidence is constructed using Equation 5.5. Using these samples, related samples for the Bayes factor are obtained. The histogram of Bayes factor samples is shown in Figure 5.7.

Three Gaussian functions:

In this case, the metamodel is constructed as follows:

$$\sum_{i=1}^{3} \rho_i \phi(\mu_{\theta_i}, \sigma_{\theta_i}) \phi(\mu_{\sigma_i}, \sigma_{\sigma_i}) \tag{5.10}$$



(a) Three dimensional plot of the metamodel

(b) Contour plot of the metamodel (dash line) on contour plot of the true model

Figure 5.6: One Gaussian Metamodel for model 2

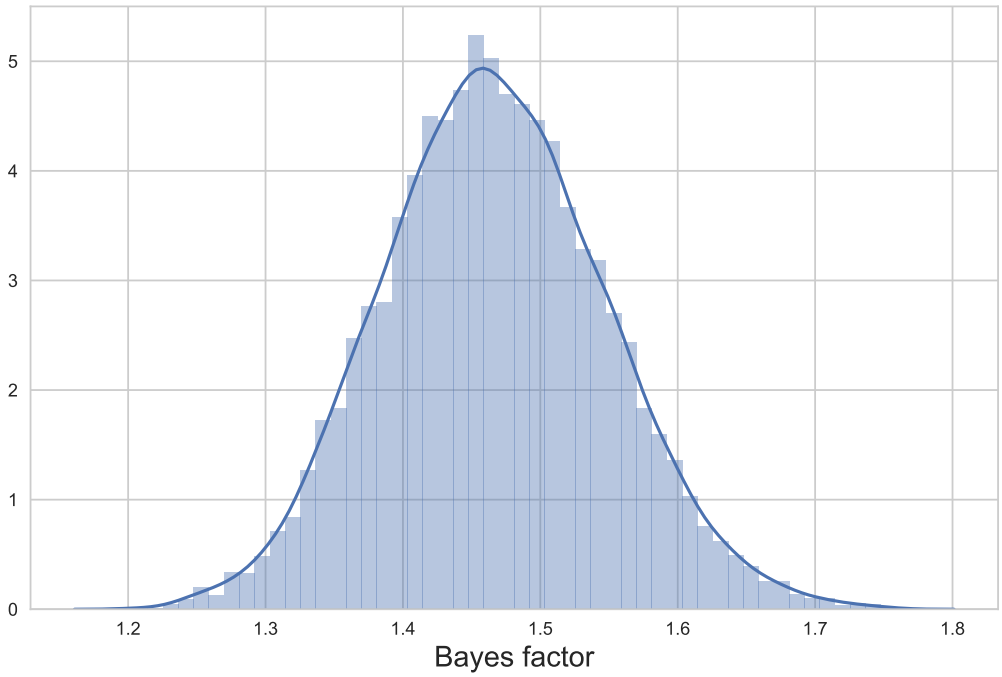
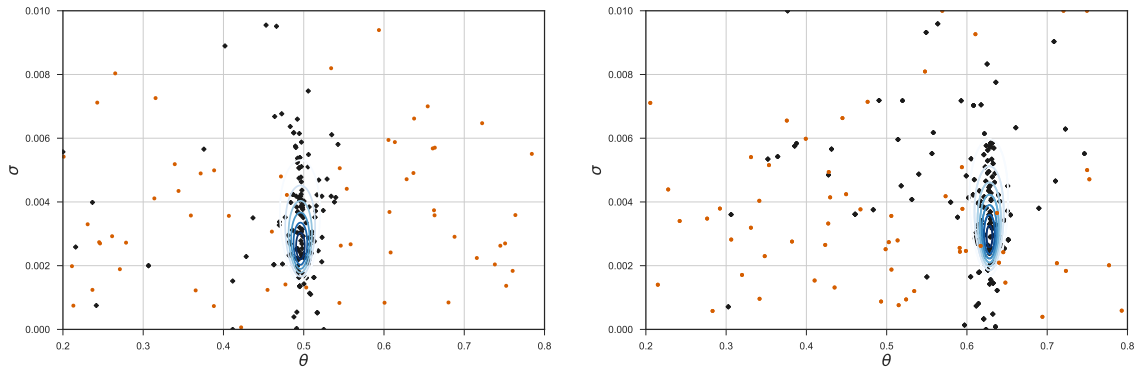


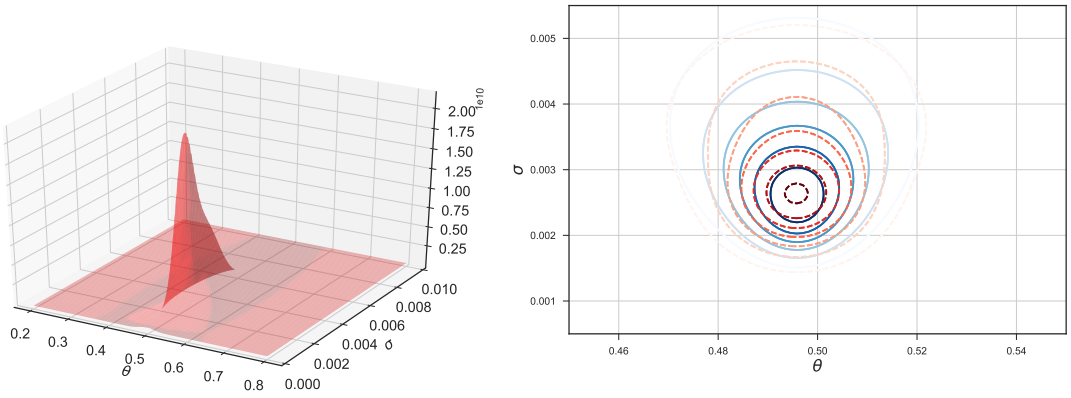
Figure 5.7: Bayes factor histogram when only one Gaussian function is used for the metamodel

Using this metamodel and considering the standard deviation of the likelihood as a free parameter, sixteen parameters should be updated (i.e.,ρ_1, ρ_2, ρ_3, μ_{θ_1}, μ_{θ_2}, μ_{θ_3}, σ_{θ_1}, σ_{θ_2}, σ_{θ_3}, μ_{σ_1}, μ_{σ_2}, μ_{σ_3}, σ_{σ_1}, σ_{σ_2}, σ_{σ_3}, and σ_{lik}). To fit this surrogate model to the true model, 60 initial samples were selected. Then, using the explained sampling



(a) Obtained samples for Model 1 (b) Obtained samples for Model 2

Figure 5.8: Obtained samples for two models

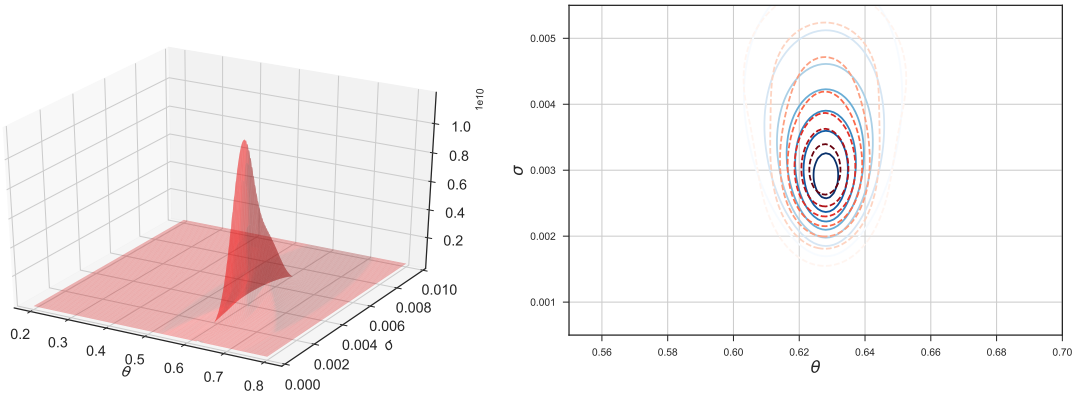


(a) Three dimensional plot of the metamodel (b) Contour plot of the metamodel (dash line) on contour plot of the true model

Figure 5.9: Three Gaussian Metamodel for model 1

strategy in chapter 4, new samples are added. For the first model, 218 samples are obtained and for the second model, 223 samples are obtained. This is almost four times of the number of samples which are needed when only one Gaussian is used. Figure 5.8 shows the obtained samples for each model.

As it is shown, in Figures 5.9 and 5.10, the three dimensional plots indicate skewness in the shape of the metamodel. Also their related contour plots show the better match in comparison with what is obtained when only one Gaussian function has been used as the metamodel. Using the chains, samples for evidence are obtained. Model evidence samples for both models are used to obtain the samples for the Bayes



(a) Three dimensional plot of the metamodel

(b) Contour plot of the metamodel (dash line) on contour plot of the true model

Figure 5.10: One Gaussian Metamodel for model 2

factor. Figure 5.11 shows the histogram for these samples. Previously, the numerical results indicated that the Bayes factor is a number between $[1.789, 1.826]$. These results are obtained by implementation of the Vegas algorithm using 10 iterations with a maximum number of 1000 integrand evaluations per iteration. Considering this interval, the histogram shown in Figure 5.11 successfully shows an estimation of the Bayes factor with only about 200 samples from each model.

5.4 Concluding remarks

In this chapter, it was shown that GMM models can be used as a general metamodel to model the EDF. Results of an example solved in this section showed, using the Vegas Monte Carlo integration, several thousand EDF evaluations are needed while in the case of using one Gaussian function almost 50 samples, and in the case of using three Gaussian functions almost 200 samples are needed. However, when only one Gaussian function is used for the metamodel skewness in the EDF cannot be captured. This creates more error compared with when more Gaussian functions are used. The challenge with implementation of many Gaussian functions is a greater number of parameters must be considered in the metamodel. This enforces taking

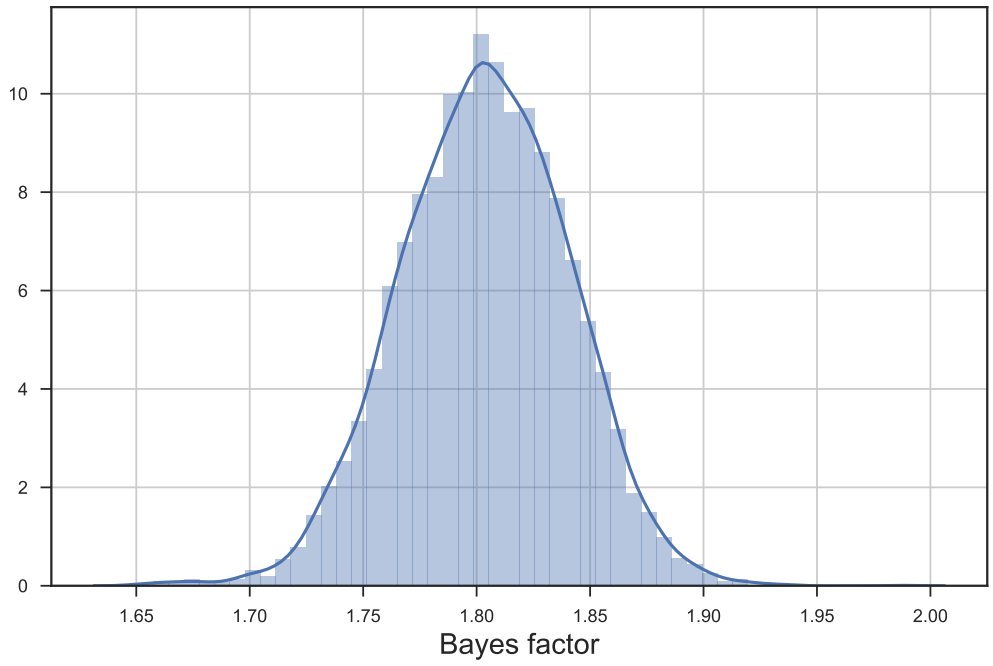


Figure 5.11: Bayes factor histogram when only one Gaussian function is used for the metamodel

more samples from EDF, and increases the computational cost.

Chapter 6

Concluding Remarks

> "Errors using inadequate data are much less than those using no data at all."
>
> *Charles Babbage*

Computationally expensive models are used intensively in science and engineering for design, optimization, uncertainty quantification, prediction, etc. In many cases, several models might be suggested to describe a particular phenomena or system. The person who uses the model should select the best model (or family of models) in light of any available data and his or her experience. One of the common methods used to select a model among different candidates is Bayes factor test. The Bayes factor is obtained from the ratio of the Bayesian evidence of two compared models and represents the probability ratio between the models considering both data and analysts experience. The Bayesian evidence is obtained from an integration over the entire parameter domain, which usually requires implementation of Monte Carlo integration methods. These methods work based on the law of large numbers, where many samples are required to obtain an accurate estimation of the Bayesian evidence. Therefore, these methods require hundreds or thousands of evaluations of the computational model describing the system. This is not feasible when a computationally expensive model is used.

One approach to overcome this high computational demands is replacing the computationally expensive model with a surrogate model. A surrogate model, or meta-

model, is an approximation for a long-running model. Different metamodels are proposed in the literature, and some of them are discussed in Chapter 1. A metamodel is selected depending on the design landscape, objective of the problem, and the inherent limitations in the original model.

The hypothesis for this work is that a PDF, or a summation of them as a metamodel, can be used to obtain the Bayes factor. In other words, the original model is not replaced directly by a metamodel. Despite being computationally expensive, the original model is used to construct a Bayesian model. Then an integrand, which is used to obtain Bayesian evidence, is modeled by the metamodel. The mathematical formulation of this idea is presented in Chapter 2. In the this chapter and Chapter 3, some examples are presented that show the implementation of the proposed technique. One of the challenges with all metamodeling techniques is the sampling strategy for selecting adaptive samples. While the proposed method can use any of the proposed sampling techniques available in the literature for metamodeling, Chapter 4 suggests using predictive posterior samples in predefined guide points to select a new adaptive sample. In the method used, not only is a new sample obtained based on the previous metamodel, but also guide points' location evolves in each run. This evolutionary sampling strategy is successfully tested on several examples. Finally, in Chapter 5 a more general metamodel, which is actually a mixture Gaussian model (GMM), is implemented in the proposed method. It is shown that the metamodel is capable of modeling the skewed EDFs.

The model of a simply supported beam with a movable support was used to explore the capabilities of the technique. The proposed method was successfully implemented to estimate the Bayes factor between two competing models. The proposed method required only 50 evaluations of the structural model while traditional Monte carlo methods required 3000. Details of this example are discussed in Page 46.

6.1 Future work

In some practical problems, due to lack of knowledge, the prior and likelihood are located far from each other in the parameters' domain. In other words, imagine the case that the domains of the integration to calculate the Bayes factor are set based on boundaries obtained from the prior, but the high probability region of the posterior is located out of the region defined by these boundaries. The region of the EDF which would be modelled with the proposed method is similar to a PDF's tail. Fitting a PDF to this region is not easy since all of the points have very low value and the changes between them is very small numbers. However, the proposed method has the potential to be helpful in two ways: 1) the modifying term for the scale factor, which is obtained from metamodel's CDF and is usually a number close to one, will be a small number close to zero because it is in the tail of the metamodel. This implies that the prior might not have been selected correctly. In other words, the reason that the Bayesian evidence is a small number for the computationally expensive model, could be inappropriate choice of the prior. 2) When the high probability region of the metamodel is located in a region outside of predefined boundaries of integration, could be used as a way to identify that the boundaries (or the prior) should be modified in a way that contains the metamodel high probability region.

In this research there is still room to propose new metamodels which are flexible to model different EDF shapes. Here, the application of GMM is suggested; however, using GMM has its own challenges. One of the challenges is that by increasing the numbers of Gaussians the number of parameters of the metamodel are increased by, at least, a factor of three. Therefore, by increasing the number of Gaussians many more samples are likely to be needed to fit the metamodel, which is not always feasible when the original model is computationally expensive.

Finally, the peak of the EDF is usually very sharp. The proposed sampling strategy shows a great performance in a low dimensional problem with a sharp peaks.

However, there is still room for future research to explore the capabilities of the method in high dimensional problems. Furthermore, other sampling strategies could be investigated to study their competence in implementation of the proposed meta-modeling technique.

References

[1] Paul L Meyer. Introductory probability and statistical applications. Technical report, 1970.

[2] Tsu T Soong. *Fundamentals of probability and statistics for engineers*. John Wiley & Sons, 2004.

[3] Jack R Benjamin and C Allin Cornell. *Probability, statistics, and decision for civil engineers*. Courier Corporation, 2014.

[4] Saeed Ghahramani. *Fundamentals of Probability: With Stochastic Processes*. CRC Press, 2015.

[5] Felipe Pamplona Mariano, L d Q Moreira, A d Silveira-Neto, Carlos Bettencourt da Silva, and JCF Pereira. A new incompressible navier-stokes solver combining fourier pseudo-spectral and immersed boundary methods. *Computer Modeling in Engineering & Sciences(CMES)*, 59(2):181–216, 2010.

[6] Z-H Han, K-S Zhang, W-P Song, and Z-D Qiao. Optimization of active flow control over an airfoil using a surrogate-management framework. *Journal of aircraft*, 47(2):603–612, 2010.

[7] CC Wong, TA Dean, and J Lin. Incremental forming of solid cylindrical components using flow forming principles. *Journal of Materials Processing Technology*, 153:60–66, 2004.

[8] G Buffa, A Ducato, and L Fratini. Numerical procedure for residual stresses prediction in friction stir welding. *Finite elements in analysis and design*, 47(4):470–476, 2011.

[9] Antony Jameson. Computational aerodynamics for aircraft design. *Science(Washington)*, 245(4916):361–371, 1989.

[10] Yew S Ong, Prasanth B Nair, and Andrew J Keane. Evolutionary optimization of computationally expensive problems via surrogate modeling. *AIAA journal*, 41(4):687–696, 2003.

[11] FJT Floris, MD Bush, Maarten Cuypers, F Roggero, and Anne Randi Syversveen. Methods for quantifying the uncertainty of production forecasts: a comparative study. *Petroleum Geoscience*, 7(S):S87–S96, 2001.

[12] Serhat Hosder, Robert Walters, and Michael Balch. Efficient uncertainty quantification applied to the aeroelastic analysis of a transonic wing. In *46th AIAA Aerospace Sciences Meeting and Exhibit*, page 729, 2008.

[13] Gary Blau, Michael Lasinski, Seza Orcun, Shuo-Huan Hsu, Jim Caruthers, Nicholas Delgass, and Venkat Venkatasubramanian. High fidelity mathematical model building with experimental data: A bayesian approach. *Computers & Chemical Engineering*, 32(4):971–989, 2008.

[14] Chad Lieberman, Karen Willcox, and Omar Ghattas. Parameter and state model reduction for large-scale statistical inverse problems. *SIAM Journal on Scientific Computing*, 32(5):2523–2542, 2010.

[15] Mojtaba Ghaedi, Mohsen Masihi, Zoltán E Heinemann, and Mohammad Hossein Ghazanfari. History matching of naturally fractured reservoirs based on the recovery curve method. *Journal of Petroleum Science and Engineering*, 126:211–221, 2015.

[16] Chunfeng Ma, Xin Li, Claudia Notarnicola, Shuguo Wang, and Weizhen Wang. Uncertainty quantification of soil moisture estimations based on a bayesian probabilistic inversion. *IEEE Transactions on Geoscience and Remote Sensing*, 55(6):3194–3207, 2017.

[17] Christian Soize. Uncertainty quantification in computational structural dynamics and vibroacoustics. In *Uncertainty Quantification*, pages 155–216. Springer, 2017.

[18] Mark A Krasnosel'skii and Aleksei V Pokrovskii. *Systems with hysteresis*. Springer Science & Business Media, 2012.

[19] David Jiles. *Introduction to magnetism and magnetic materials*. CRC press, 2015.

[20] ID Mayergoyz. The classical preisach model of hysteresis. In *Mathematical Models of Hysteresis*, pages 1–63. Springer, 1991.

[21] E Cardelli, M Carpentieri, A Faba, and G Finocchio. Modeling of hysteresis in magnetic multidomains. *Physica B: Condensed Matter*, 435:62–65, 2014.

[22] E Cardelli, A Faba, A Laudani, GM Lozito, F Riganti Fulginei, and A Salvini. A neural-fem tool for the 2-d magnetic hysteresis modeling. *Physica B: Condensed Matter*, 486:111–115, 2016.

[23] Carlos G Dávila, Pedro P Camanho, and Albert Turon. Effective simulation of delamination in aeronautical structures using shells and cohesive elements. *Journal of Aircraft*, 45(2):663–672, 2008.

[24] A Turon, PP Camanho, J Costa, and J Renart. Accurate simulation of delamination growth under mixed-mode loading using cohesive elements: definition of interlaminar strengths and elastic stiffness. *Composite Structures*, 92(8):1857–1864, 2010.

[25] Kexin Chang, Yuki Ohmura, Osamu Watanabe, and Akihiro Matsuda. J integral in elasto-plasticity by path integral method and virtual crack extension method in 2-dimensional problem. In *ASME 2013 Pressure Vessels and Piping Conference*, pages V06AT06A013–V06AT06A013. American Society of Mechanical Engineers, 2013.

[26] Vinay K Goyal, Eric R Johnson, and Carlos G Davila. Irreversible constitutive law for modeling the delamination process using interfacial surface discontinuities. *Composite Structures*, 65(3):289–305, 2004.

[27] G Alfano and MA Crisfield. Finite element interface models for the delamination analysis of laminated composites: mechanical and computational issues. *International journal for numerical methods in engineering*, 50(7):1701–1736, 2001.

[28] A Turon, Pedro Ponces Camanho, J Costa, and CG Dávila. A damage model for the simulation of delamination in advanced composites under variable-mode loading. *Mechanics of Materials*, 38(11):1072–1089, 2006.

[29] F Shen, KH Lee, and TE Tay. Modeling delamination growth in laminated composites. *Composites Science and Technology*, 61(9):1239–1251, 2001.

[30] Xiaole Li and Jiye Chen. A highly efficient prediction of delamination migration in laminated composites using the extended cohesive damage model. *Composite Structures*, 160:712–721, 2017.

[31] S Dey, T Mukhopadhyay, and S Adhikari. Metamodel based high-fidelity stochastic analysis of composite laminates: A concise review with critical comparative assessment. *Composite Structures*, 2017.

[32] Jake VanderPlas. Frequentism and bayesianism: a python-driven primer. *arXiv preprint arXiv:1411.5018*, 2014.

[33] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? does it matter? *Structural Safety*, 31(2):105–112, 2009.

[34] Brian Dennis. Discussion: should ecologists become bayesians? *Ecological Applications*, 6(4):1095–1103, 1996.

[35] M Jésus Bayarri and James O Berger. The interplay of bayesian and frequentist analysis. *Statistical Science*, pages 58–80, 2004.

[36] I Boulkaibet, T Marwala, L Mthembu, MI Friswell, and S Adhikari. Sampling techniques in bayesian finite element model updating. In *Topics in Model Validation and Uncertainty Quantification, Volume 4*, pages 75–83. Springer, 2012.

[37] I Boulkaibet, L Mthembu, T Marwala, MI Friswell, and S Adhikari. Finite element model updating using the shadow hybrid monte carlo technique. In *Special Topics in Structural Dynamics, Volume 6*, pages 489–498. Springer, 2013.

[38] I Boulkaibet, L Mthembu, T Marwala, MI Friswell, and S Adhikari. Finite element model updating using the shadow hybrid monte carlo technique. *Mechanical Systems and Signal Processing*, 52:115–132, 2015.

[39] James L Beck and Siu-Kui Au. Bayesian updating of structural models and reliability using markov chain monte carlo simulation. *Journal of Engineering Mechanics*, 128(4):380–391, 2002.

[40] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.

[41] RE Melchers. Importance sampling in structural systems. *Structural safety*, 6(1):3–10, 1989.

[42] Sai Hung Cheung and James L Beck. Bayesian model updating using hybrid monte carlo simulation with application to structural dynamic models with many uncertain parameters. *Journal of engineering mechanics*, 135(4):243–255, 2009.

[43] Bo Cai, Renate Meyer, and François Perron. Metropolis–hastings algorithms with adaptive proposals. *Statistics and Computing*, 18(4):421–433, 2008.

[44] Z Jiang and R Christenson. A comparison of 200 kn magneto-rheological damper models for use in real-time hybrid simulation pretesting. *Smart Materials and Structures*, 20(6):065011, 2011.

[45] Malcolm R Forster. Key concepts in model selection: Performance and generalizability. *Journal of mathematical psychology*, 44(1):205–231, 2000.

[46] Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

[47] Kiyoshi Yamaoka, Terumichi Nakagawa, and Toyozo Uno. Application of akaike's information criterion (aic) in the evaluation of linear pharmacokinetic equations. *Journal of pharmacokinetics and biopharmaceutics*, 6(2):165–175, 1978.

[48] KUNIO Tanabe. Statistical regularization of a noisy ill-conditioned system of linear equations by akaike's information criterion. *Research Memo*, (60), 1974.

[49] Hirotugu Akaike. Akaike's information criterion. In *International Encyclopedia of Statistical Science*, pages 25–25. Springer, 2011.

[50] Hamparsum Bozdogan. Model selection and akaike's information criterion (aic): The general theory and its analytical extensions. *Psychometrika*, 52(3):345–370, 1987.

[51] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[52] M Stone. Comments on model selection criteria of akaike and schwarz. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 276–278, 1979.

[53] David L Weakliem. A critique of the bayesian information criterion for model selection. *Sociological Methods & Research*, 27(3):359–397, 1999.

[54] Kenneth P Burnham and David R Anderson. *Model selection and multimodel inference: a practical information-theoretic approach*. Springer Science & Business Media, 2002.

[55] Yuhong Yang. Can the strengths of aic and bic be shared? a conflict between model indentification and regression estimation. *Biometrika*, 92(4):937–950, 2005.

[56] Scott I Vrieze. Model selection and psychological theory: a discussion of the differences between the akaike information criterion (aic) and the bayesian information criterion (bic). *Psychological methods*, 17(2):228, 2012.

[57] Robert E Kass and Adrian E Raftery. Bayes factors. *Journal of the american statistical association*, 90(430):773–795, 1995.

[58] Adrian E Raftery. Approximate bayes factors and accounting for model uncertainty in generalised linear models. *Biometrika*, 83(2):251–266, 1996.

[59] Adrian FM Smith and David J Spiegelhalter. Bayes factors and choice criteria for linear models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 213–220, 1980.

[60] Jennifer A Hoeting, David Madigan, Adrian E Raftery, and Chris T Volinsky. Bayesian model averaging: a tutorial. *Statistical science*, pages 382–401, 1999.

[61] Harold Jeffreys. *The theory of probability*. OUP Oxford, 1998.

[62] Edwin T Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.

[63] Andreas Berg, Renate Meyer, and Jun Yu. Deviance information criterion for comparing stochastic volatility models. *Journal of Business & Economic Statistics*, 22(1):107–120, 2004.

[64] David Posada and Thomas R Buckley. Model selection and model averaging in phylogenetics: advantages of akaike information criterion and bayesian approaches over likelihood ratio tests. *Systematic biology*, 53(5):793–808, 2004.

[65] Martin Sewell. Structural risk minimization. 2008.

[66] Cong Han and Bradley P Carlin. Markov chain monte carlo methods for computing bayes factors: A comparative review. *Journal of the American Statistical Association*, 96(455):1122–1132, 2001.

[67] Ronald L Iman. Latin hypercube sampling. *Encyclopedia of Quantitative Risk Analysis and Assessment*, 2008.

[68] Michael Stein. Large sample properties of simulations using latin hypercube sampling. *Technometrics*, 29(2):143–151, 1987.

[69] Felipe AC Viana. Things you wanted to know about the latin hypercube design and were afraid to ask. In *10th World Congress on Structural and Multidisciplinary Optimization*, pages 1–9. sn, 2013.

[70] Jayant R Kalagnanam and Urmila M Diwekar. An efficient sampling technique for off-line quality control. *Technometrics*, 39(3):308–319, 1997.

[71] ZHANG RUNCHU and WANG ZHAOJUN. Uniform design sampling and its fine properties [j]. *Chinese Journal of Applied Probability and Statisties*, 4:000, 1996.

[72] Mark E Johnson, Leslie M Moore, and Donald Ylvisaker. Minimax and maximin distance designs. *Journal of statistical planning and inference*, 26(2):131–148, 1990.

[73] Alexander IJ Forrester and Andy J Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1):50–79, 2009.

[74] Ferdinand Pierre Beer, Elwood Russell Johnston, and John T.. DeWolf. *Statics and mechanics of materials*. McGraw-Hill Education, 2017.

[75] Richard F Gunst. Response surface methodology: process and product optimization using designed experiments, 1996.

[76] Russell R Barton. Simulation metamodels. In *Simulation Conference Proceedings, 1998. Winter*, volume 1, pages 167–174. IEEE, 1998.

[77] Peter Lancaster and Kes Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of computation*, 37(155):141–158, 1981.

[78] Piotr Breitkopf, Hakim Naceur, Alain Rassineux, and Pierre Villon. Moving least squares response surface approximation: Formulation and metal forming applications. *Computers & Structures*, 83(17):1411–1428, 2005.

[79] Margaret A Oliver and Richard Webster. Kriging: a method of interpolation for geographical information systems. *International Journal of Geographical Information System*, 4(3):313–332, 1990.

[80] Noel Cressie. Spatial prediction and ordinary kriging. *Mathematical Geology*, 20(4):405–421, 1988.

[81] Jack PC Kleijnen. Kriging metamodeling in simulation: A review. *European journal of operational research*, 192(3):707–716, 2009.

[82] Timothy W Simpson, Timothy M Mauery, John J Korte, and Farrokh Mistree. Kriging models for global approximation in simulation-based multidisciplinary design optimization. *AIAA journal*, 39(12):2233–2241, 2001.

[83] Boris A Zárate and Juan M Caicedo. Finite element model updating: Multiple alternatives. *Engineering Structures*, 30(12):3724–3730, 2008.

[84] Michael Friswell and John E Mottershead. *Finite element model updating in structural dynamics*, volume 38. Springer, 1995.

[85] SV Modak, TK Kundra, and BC Nakra. Prediction of dynamic characteristics using updated finite element models. *Journal of Sound and Vibration*, 254(3):447–467, 2002.

[86] SD Rajan. Sizing, shape, and topology design optimization of trusses using genetic algorithm. *Journal of Structural Engineering*, 121(10):1480–1487, 1995.

[87] Christian P Robert and George Casella. *Monte Carlo statistical methods*, volume 319. Citeseer, 2004.

[88] D Andrew Brown, Arvind K Saibaba, and Sarah Vallélian. Computationally efficient markov chain monte carlo methods for hierarchical bayesian inverse problems. *arXiv preprint arXiv:1609.07180*, 2016.

[89] Mohammed A El-Beltagy and AJ Keane. Metalmodeling techniques for evolutionary optimization of computationally expensive problems: promises and limitations. 1999.

[90] Martin Meckesheimer, Andrew J Booker, Russell R Barton, and Timothy W Simpson. Computationally inexpensive metamodel assessment strategies. *AIAA journal*, 40(10):2053–2060, 2002.

[91] Ramin Madarshahian, Juan M Caicedo, and Diego Arocha Zambrana. Evaluation of a time reversal method with dynamic time warping matching function for human fall detection using structural vibrations. In *Model Validation and Uncertainty Quantification, Volume 3*, pages 171–176. Springer, 2014.

[92] Art B Owen. Orthogonal arrays for computer experiments, integration and visualization. *Statistica Sinica*, 2(2):439–452, 1992.

[93] Peter W Glynn and Donald L Iglehart. Importance sampling for stochastic simulations. *Management Science*, 35(11):1367–1392, 1989.

[94] Songqing Shan and G Gary Wang. Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. *Structural and Multidisciplinary Optimization*, 41(2):219–241, 2010.

[95] G Gary Wang and S Shan. Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical Design*, 129(4):370–380, 2007.

[96] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43, 2003.

[97] Daniel F Waggoner and Tao Zha. A gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, 28(2):349–366, 2003.

[98] B Gaspar, AP Teixeira, and C Guedes Soares. Adaptive surrogate model with active refinement combining kriging and a trust region method. *Reliability Engineering & System Safety*, 165:277–291, 2017.

[99] Juan D Ocampo, Nathan E Crosby, Harry R Millwater, Elias L Anagnostou, Stephen J Engel, John S Madsen, and Karin W Engel. Probabilistic damage tolerance for aviation fleets using a kriging surrogate model. In *19th AIAA Non-Deterministic Approaches Conference*, page 1567, 2017.

[100] Motahareh Saadatpour, Abbas Afshar, and John Eric Edinger. Meta-model assisted 2d hydrodynamic and thermal simulation model (ce-qual-w2) in deriving optimal reservoir operational strategy in selective withdrawal scheme. *Water Resources Management*, 31(9):2729–2744, 2017.

[101] Hector A Jensen, C Esse, V Araya, and C Papadimitriou. Implementation of an adaptive meta-model for bayesian finite element model updating in time domain. *Reliability Engineering & System Safety*, 160:174–190, 2017.

[102] Ramin Madarshahian and Juan M Caicedo. Reducing mcmc computational cost with a two layered bayesian approach. In *Model Validation and Uncertainty Quantification, Volume 3*, pages 291–297. Springer, 2015.

[103] William H Press and Glennys R Farrar. Recursive stratified sampling for multidimensional monte carlo integration. *Computers in Physics*, 4(2):190–195, 1990.

[104] G Peter Lepage. A new algorithm for adaptive multidimensional integration. *Journal of Computational Physics*, 27(2):192–203, 1978.

[105] G Peter Lepage. Vegas-an adaptive multi-dimensional integration program. Technical report, 1980.

[106] Man-Suk Oh and James O Berger. Adaptive importance sampling in monte carlo integration. *Journal of Statistical Computation and Simulation*, 41(3-4):143–168, 1992.

[107] Russel E Caflisch. Monte carlo and quasi-monte carlo methods. *Acta numerica*, 7:1–49, 1998.

[108] Dirk Ostwald, Ludger Starke, and Ralph Hertwig. A normative inference approach for optimal sample sizes in decisions from experience. *Frontiers in psychology*, 6, 2015.

[109] Douglas Reynolds. Gaussian mixture models. *Encyclopedia of biometrics*, pages 827–832, 2015.

[110] Anand Patil, David Huard, and Christopher J Fonnesbeck. Pymc: Bayesian stochastic modelling in python. *Journal of statistical software*, 35(4):1, 2010.

[111] Ramin Madarshahian and Juan M Caicedo. Metamodeling of model evidence. In *Model Validation and Uncertainty Quantification, Volume 3*, pages 307–313. Springer, 2016.

[112] Christian Robert and George Casella. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

[113] Andreĭ Kolmogorov. Foundations of the theory of probability. 1950.

[114] Saeed Eftekhar Azam and Stefano Mariani. Dual estimation of partially observed nonlinear structural systems: A particle filter approach. *Mechanics Research Communications*, 46:54–61, 2012.

[115] S Eftekhar Azam, M Bagherinia, and S Mariani. Stochastic system identification via particle and sigma-point kalman filtering. *Scientia Iranica*, 19(4):982–991, 2012.

[116] Andrew Fitzgibbon, Maurizio Pilu, and Robert B Fisher. Direct least square fitting of ellipses. *IEEE Transactions on pattern analysis and machine intelligence*, 21(5):476–480, 1999.

[117] Derek York. Least squares fitting of a straight line with correlated errors. *Earth and planetary science letters*, 5:320–324, 1968.

[118] Cuthbert Daniel and Fred S Wood. *Fitting equations to data: computer analysis of multifactor data*. John Wiley & Sons, Inc., 1999.

[119] Chain Monte Carlo. Markov chain monte carlo and gibbs sampling. *Lecture Notes for EEB 581*, 2004.

[120] Adrian E Raftery and Steven M Lewis. Implementing mcmc. *Markov chain Monte Carlo in practice*, pages 115–130, 1996.

[121] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

[122] Kenneth Price, Rainer M Storn, and Jouni A Lampinen. *Differential evolution: a practical approach to global optimization*. Springer Science & Business Media, 2006.

[123] David Hitchcock. Lecture notes in introduction to bayesian data analysis (stat 535). 2014.

Appendix A

A sample python code

The python code of the example shown at the page 61 is presented as a sample python code:

```
# -*- coding: utf-8 -*-
"""
@author: madarshahian
"""

import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pymc as pm
from scipy.stats import norm
from scipy.stats import skew
from scipy import optimize as opt
from scipy.optimize import differential_evolution
sns.set(color_codes=True)
sns.set_color_codes("colorblind")
sns.set_style("ticks",{"axes.grid": True})
plt.close("all")
from matplotlib import rcParams
rcParams.update({'figure.autolayout': True})
np.random.seed(1000)
#%% creating the data of structural example
loading=np.array([[10000,0],[10000,0],[0,20000],[0,20000]])#The first one is p
#(N) and the second one is M (N.m)
theta_real=.5
l=10.0##m total length of the beam
E=200e9#N/m
I=1e-4##m^4
P=loading[:,0]
M=loading[:,1]
```

```
d=(l**2./(E*I))*(P*l*theta_real**2/3.+M*(theta_real**2/6.+theta_real/3.))\
+np.random.normal(0,0.005,len(loading))
#%% Bayesian Model of the structure
#priors
theta_down=0.2
theta_up=0.8
theta=pm.Uniform('theta',theta_down,theta_up)#prior is limited
sigma_up=0.01
sigma_down=0.000001
sigma_like=pm.Gamma('sigmalike',2,1/0.002)#in stats scipy the shape factor is
#reverese
#Model
@pm.deterministic #true model
def disp(theta=theta,loading=loading,l=l,E=E,I=I):
    P=loading[:,0]
    M=loading[:,1]
    return (l**2./(E*I))*(P*l*theta**2/3.+M*(theta**2/6.+theta/3.)) #True model
    #return (l**2./(E*I))*(P*l*theta**3/3.+M/2.*theta**2)#Bad model
#likelihood
obs=pm.Normal('obs',disp,1./sigma_like**2,observed=True,value=d)
#%% evidence integrand as the true model
def true_model(theta_model1=1/3.,sigma=0.002,loading=loading):
    theta.set_value(theta_model1)
    sigma_like.set_value(sigma)
    return np.exp(obs.logp)*np.exp(theta.logp)*np.exp(sigma_like.logp)
#%% Plot countor
start_theta=theta_down
stop_theta=theta_up
num_theta=500.
start_sigma=sigma_down
stop_sigma=sigma_up
num_sigma=500
theta_plt = np.linspace(start_theta,stop_theta,num_theta)
sigma_plt = np.linspace(start_sigma, stop_sigma, num_sigma)
T, S = np.meshgrid(theta_plt, sigma_plt)
EDF2=np.empty([len(T)*len(S.T),1])
count=0
for item in zip(T.reshape([len(T)*len(S.T),1]),S.reshape([len(T)*len(S.T),1])):
    EDF2[count]=true_model(item[0],item[1])
    count+=1
EDF2[np.isnan(EDF2)] = 0#convert nan to zero
```

```
71  Area=sum(EDF2)*(stop_theta-start_theta)/num_theta*(stop_sigma-start_sigma)/\
72          num_sigma
73  print "Evidence is:",Area
74  #%%
75  EDF3=EDF2.reshape([len(T),len(S.T)])
76  plt.figure(0)
77  CS = plt.contour(T, S, EDF3,cmap='Blues')
78  plt.ylim([0,sigma_up])
79  plt.xlabel(r'$\theta$',fontsize=20)
80  plt.ylabel(r'$\sigma$',fontsize=20)
81  #%%
82  T_lin=T.reshape([len(T)*len(S.T),1])[:,0]
83  S_lin=S.reshape([len(T)*len(S.T),1])[:,0]
84  EDF_lin=EDF2[:,0]
85  EDF_lin[np.isnan(EDF_lin)] = 0#convert nan to zero
86  del M,P,E,I,loading,l,count,d
87
88  #%%
89  fig = plt.figure(1)
90  ax = fig.add_subplot(111, projection='3d')
91  ax.plot_surface(T,S,EDF3,rstride=1,cstride=1,alpha=0.3)
92
93  for tick in ax.xaxis.get_major_ticks():
94      tick.label.set_fontsize(14)
95  for tick in ax.yaxis.get_major_ticks():
96      tick.label.set_fontsize(14)
97  for tick in ax.zaxis.get_major_ticks():
98      tick.label.set_fontsize(14)
99  ax.set_ylim([0,sigma_up])
100 ax.set_xlabel('\n'+r'$\theta$',fontsize=14)
101 ax.set_ylabel('\n'+r'$\sigma$',fontsize=14)
102 plt.savefig('figures/True_model_3d.png')
103 plt.savefig('figures/True_model_3d.pdf')
104
105 #ax.zlabel(r"$f(X,Y)$")
106 plt.show()
107 #%% Metamodeling
108 def priors(size_theta=3,size_sigma=3):
109     a=pm.Normal('Mean_in_Theta',0.5,1/.2**2,size=1)
110     a_s=pm.Gamma('Standard_deviation_in_Thata',2,1/.1**2,size=size_theta)
111     b=pm.Normal('Mean_in_Sigmalik',0,1./((.3*sigma_up)**2),size=size_sigma)
```

```
112     b_s=pm.Gamma('Standard_deviation_in_Sigmalik',2,1/(0.015*sigma_up),\
113                  size=size_sigma)
114     sf_m=[0]+[5e6]*(size_sigma-1)
115     sf=pm.Normal('SF',sf_m,[1./((3e5)**2)]*size_sigma,size=size_sigma)
116     sigma_like_met=pm.Gamma('sigma_like_met',1.1,1./1e7)
117     num_gaussian=size_theta
118     return a,a_s,b,b_s,sf,sigma_like_met,size_theta,size_sigma,num_gaussian
119 a,a_s,b,b_s,sf,sigma_like_met,size_theta,size_sigma,num_gaussian=priors()
120 @pm.potential(plot=True)
121 def pot(sf_all=sf):
122     if sf_all[0]<sf_all[1] and sf_all[1]<sf_all[2]:
123         return 0. ######## No penalty applied if conditions are met
124     else:
125         return -np.inf ###### Infinite potential applied if not met
126
127 #%% Initial samples
128 num_i=60
129 def initial_samples(num_i=num_i):
130     ''' Adds INITIAL_SAMPLES and ADDED_PER_STEP samples '''
131     samples_theta=np.empty(num_i)
132     samples_sigma=np.empty(num_i)
133     samples_edf=np.empty(num_i)
134     for i in range(num_i):#Initial samples are created here
135         samples_theta[i]=theta.random()
136         if samples_theta[i]<theta_down:
137             samples_theta[i]=theta_down
138         elif samples_theta[i]>theta_up:
139             samples_theta[i]=theta_up
140         samples_sigma[i]=sigma_like.random()
141         if samples_sigma[i]<sigma_down:
142             samples_sigma[i]=sigma_down
143         elif samples_sigma[i]>sigma_up:
144             samples_sigma[i]=sigma_up
145         samples_edf[i]=true_model(samples_theta[i],samples_sigma[i])
146     samples_edf[np.isnan(samples_edf)] = 0#convert nan to zero
147     return samples_theta,samples_sigma,samples_edf,num_i
148 Xi,Yi,Zi,add_sample_indx=initial_samples(num_i=num_i)
149 plt.plot(Xi,Yi,'ro')
150
151 #%%Defining GPS algorithm
152 x_pre=np.array([])
```

```
153 y_pre=np.array([])
154 sd_predict=np.array([])
155 def GPS(num_gps=20,num_eliminated=10,x_pre=x_pre,y_pre=y_pre,sd_predict=\
156         sd_predict,x_pre_lim=[theta_down,theta_up],\
157                                 y_pre_lim=[sigma_down,sigma_up]):
158     if len(sd_predict)==0:#only for the first run
159         x_pre=np.random.uniform(x_pre_lim[0],x_pre_lim[1],[num_gps,1])
160         y_pre=np.random.uniform(y_pre_lim[0],y_pre_lim[1],[num_gps,1])
161     else:
162         if sd_predict.shape[-1]==1:
163             sd_predict=sd_predict[:,0]
164         x_pre=x_pre[sd_predict.argsort()]#sort it from the smallest to largest
165         y_pre=y_pre[sd_predict.argsort()]#sort it from the smallest to largest
166         x_pre[:num_eliminated/3]=np.random.uniform(x_pre_lim[0],x_pre_lim[1],\
167               [num_eliminated/3,1])
168         y_pre[:num_eliminated/3]=np.random.uniform(y_pre_lim[0],y_pre_lim[1],\
169               [num_eliminated/3,1])
170         x_pre[num_eliminated/3:num_eliminated]=\
171               np.random.normal(x_pre[num_eliminated:].mean(),\
172                                x_pre[num_eliminated:].std(),\
173                                     [num_eliminated-num_eliminated/3,1])
174         y_pre[num_eliminated/3:num_eliminated]=\
175               np.random.normal(y_pre[num_eliminated:].mean(),\
176                                y_pre[num_eliminated:].std(),\
177                                     [num_eliminated-num_eliminated/3,1])
178         x_pre[x_pre<x_pre_lim[0]]=x_pre_lim[0]
179         y_pre[y_pre<y_pre_lim[0]]=y_pre_lim[0]
180         x_pre[x_pre>x_pre_lim[1]]=x_pre_lim[1]
181         y_pre[y_pre>y_pre_lim[1]]=y_pre_lim[1]
182     return x_pre,y_pre
183 #%%
184 def met_model(x_samples,y_samples,z_samples,\
185                predict=False,x_pre=x_pre,y_pre=y_pre,sd_predict=sd_predict,\
186                x_pre_lim=[theta_down,theta_up],y_pre_lim=[sigma_down,sigma_up]):
187     @pm.deterministic
188     def met(sf=sf,a=a,a_s=a_s,b=b,b_s=b_s,inpt=\
189             np.vstack((x_samples,y_samples)).T):
190         return sum([sf[i]*norm.pdf(inpt[:,0],a,a_s[i])*norm.pdf(inpt[:,1],\
191                     b[i],b_s[i]) for i in range(num_gaussian)])
192         #return sf[0]*norm.pdf(inpt[:,0],a[0],a_s[0])*norm.pdf(inpt[:,1],b[0],\
193         #b_s[0])+sf[1]*norm.pdf(inpt[:,0],a[1],a_s[1])*norm.pdf(inpt[:,1],b[1]\
```

```
194             #,b_s[1])
195
196         obs_met=pm.Normal('obs_met',met,1./sigma_like_met**2,\
197                           observed=True,value=z_samples)
198         if predict==False:
199             model_met=pm.MCMC({'obs_met':obs_met,'sf':sf,'a':a,'a_s':a_s,\
200             'b':b,'b_s':b_s,'sigma_like_met':sigma_like_met})
201             return model_met
202         else:
203             #samples for GPS prediction
204             x_pre,y_pre=GPS(num_gps=20,num_eliminated=10,x_pre=x_pre,\
205                             y_pre=y_pre,sd_predict=sd_predict,x_pre_lim=\
206                             [theta_down,theta_up],y_pre_lim=[sigma_down,sigma_up])
207
208             @pm.deterministic
209             def met_pre(sf=sf,a=a,a_s=a_s,b=b,b_s=b_s,inpt=\
210                         np.vstack((x_pre[:,0],y_pre[:,0]))).T):
211                 return sum([sf[i]*norm.pdf(inpt[:,0],a,a_s[i])*\
212                             norm.pdf(inpt[:,1],b[i],\
213                                      b_s[i]) for i in range(size_sigma)])
214                 #return  sf[0]*norm.pdf(inpt[:,0],a[0],a_s[0])*
215                 #norm.pdf(inpt[:,1],b[0],b_s[0])+sf[1]*
216                         #norm.pdf(inpt[:,0],a[1],a_s[1])*
217                                  #norm.pdf(inpt[:,1],b[1],b_s[1])
218
219             obs_met_predict=pm.Normal('obs_met_predict',\
220                                       met_pre,1./sigma_like_met**2)
221             model_met=pm.MCMC({'obs_met_predict':obs_met_predict,'obs_met':\
222                                obs_met,'sf':sf,'a':a,'a_s':a_s,'b':b,'b_s':b_s\
223                                ,'sigma_like_met':sigma_like_met})
224             return model_met,x_pre,y_pre
225
226 x1_d=-3.0e7
227 x1_u=3.0e7
228 x2_d=.4
229 x2_u=.6
230 x3_d=1e-6
231 x3_u=5e-2
232 x4_d=-5e-1
233 x4_u=5e-1
234 x5_d=1e-6
```

```
x5_u=2e-2
x6_d=1e-6
x6_u=1.0e9

idx=num_gaussian
mm=idx-1
def MAPlog(x):
    #print x
    if x[-1]<=0 or np.less(x[2*idx:3*idx],0).any() or \
        np.less(x[4*idx:5*idx],0).any():
        return np.inf
    else:
        sf.set_value(x[0:idx])
        a.set_value(x[idx:2*idx-mm])
        a_s.set_value(x[2*idx-mm:3*idx-mm])
        b.set_value(x[3*idx-mm:4*idx-mm])
        b_s.set_value(x[4*idx-mm:5*idx-mm])
        sigma_like_met.set_value(x[-1])
    for i in range(idx-1):
        if x[i+1]-x[i]<0:
            return np.inf
    return -model_met.logp
bounds = [(x1_d,x1_u)]*num_gaussian+[(x2_d,x2_u)]*1+[(x3_d,x3_u)]*\
          num_gaussian+[(x4_d,x4_u)]*num_gaussian+[(x5_d,x5_u)]*\
                              num_gaussian+[(x6_d,x6_u)]

#%%MCMC
logmap=[]
x_best=[]
model_met=met_model(Xi,Yi,Zi,predict=False)
result = differential_evolution(MAPlog,bounds,\
                                    strategy='best1bin',popsize=15,tol=1e-10)
result.x, result.fun
print('logp before nelder-mead={:06.2f}'.format(model_met.logp))
this_x=opt.minimize(MAPlog, result.x, method='nelder-mead')
x_opt=this_x.x
x_best.append(x_opt)
print('\nlogp after nelder-mead={:06.2f}'.format(model_met.logp))
logmap.append(model_met.logp)
MAPlog(x_opt)
model_met.sample(iter=10000, burn=6000, thin=10)
```

```
print('\nlogp after MCMC={:06.2f}'.format(model_met.logp))

#pm.Matplot.plot(model_met)
#%%
a_mean=a.trace()[-100:].mean()
a_s_mean=a_s.trace()[-100:].mean(axis=0)
b_mean=b.trace()[-100:].mean(axis=0)
b_s_mean=b_s.trace()[-100:].mean(axis=0)
sf_mean=sf.trace()[-100:].mean(axis=0)

def meta_model(sf=sf_mean,a=a_mean,a_s=a_s_mean,\
               b=b_mean,b_s=b_s_mean,inpt=[1,1]):
    return sum([sf[i]*norm.pdf(inpt[0],a,a_s[i])*\
                norm.pdf(inpt[1],b[i],b_s[i]) for i in range(len(sf))])

z_met=meta_model(inpt=[T,S])
plt.figure(2)
#levels=np.arange(0.1,1.5,0.5)
#CS = plt.contour(x, y, z_met,cmap='Reds',ls=".-",levels=levels)
CS = plt.contour(T, S, z_met,cmap='Reds',ls=".-")

#plt.annotate('Metamodel from initial samples', xy=(a_mean, b_mean+3
#*b_s_mean), xytext=(1, 4),arrowprops=dict(facecolor='red', shrink=0.05),)
#%%Calculating the evidence samples and first three moments
def ev_samples(b=b,b_s=b_s,sf=sf,down_lim=0, up_lim=sigma_up):
    mf_samples=norm.cdf(up_lim,b.trace(),b_s.trace())-\
                        norm.cdf(down_lim,b.trace(),b_s.trace())#modification
                        #factores
    ev_comps=sf.trace()*mf_samples
    ev_samples=ev_comps.sum(axis=1)
    return ev_samples,ev_samples[-int(ev_samples.shape[0]/4):-10].mean(),\
ev_samples[-int(ev_samples.shape[0]/4):-10].std(),\
            skew(ev_samples[-int(ev_samples.shape[0]/4):-10])
#%%
SD_samples=[]
print("\nLoop is started:")
stopping_indx=0
stopping=np.array([1,1.])
stp_stp=[]
stp_old=np.array([0,0.])
#x_samples,y_samples,z_samples,add_sample_indx=initial_samples(num_i)
```

```
317 add_sample_indx=400
318 name4='sample_results/samples_new_'+'gaussian_GPS'+\
319 str(num_i)+'_'+str(add_sample_indx)+'.txt'
320 x_samples,y_samples,z_samples=np.loadtxt(name4)
321 all_predict=[]
322 #%%
323 x_all_gps=[]
324 y_all_gps=[]
325 #sys.exit(1)
326 while stopping_indx<3:
327     if np.sqrt((stopping[0]/0.02)**2 + (stopping[1]/0.1)**2)<np.sqrt(2):
328         stopping_indx+=1
329     else:
330         stopping_indx=0
331     a,a_s,b,b_s,sf,sigma_like_met,size_theta,size_sigma,num_gaussian=priors()
332 #     @pm.potential(plot=True)
333 #     def pot(sf_all=sf):
334 #         if sf_all[0]<sf_all[1] and sf_all[1]<sf_all[2]:
335 #             return 0. ######## No penalty applied if conditions are met
336 #         else:
337 #             return -np.inf ##### Infinite potential applied if not met
338
339     model_met=met_model(x_samples,y_samples,z_samples)
340     result = differential_evolution(MAPlog,bounds,strategy='best1bin'\
341                                     ,popsize=15,tol=1e-10)
342     result.x, result.fun
343     x_opt=result.x
344     print('logp before nelder-mead={:06.3f}'.format(model_met.logp))
345     this_x=opt.minimize(MAPlog, result.x, method='nelder-mead')
346     x_opt=this_x.x
347     x_best.append(x_opt)
348     print('\nlogp after nelder-mead={:06.3f}'.format(model_met.logp))
349     logmap.append(model_met.logp)
350     #%%
351     def met_model_fun(x=x_opt,inpt=[1,1]):
352         sf=x[0:idx]
353         a=x[idx:2*idx-mm]
354         a_s=x[2*idx-mm:3*idx-mm]
355         b=x[3*idx-mm:4*idx-mm]
356         b_s=x[4*idx-mm:5*idx-mm]
357         return sum([sf[i]*norm.pdf(inpt[0],a,a_s[i])*norm.pdf(inpt[1],\
```

```
358                         b[i],b_s[i]) for i in range(len(sf))])
359             #return sf[0]*norm.pdf(inpt[0],a[0],a_s[0])*norm.pdf(inpt[1],b[0],
360             #b_s[0])+sf[1]*norm.pdf(inpt[0],a[1],a_s[1])*
361             #norm.pdf(inpt[1],b[1],b_s[1])
362
363
364         Z_met=met_model_fun(x=x_opt,inpt=[T,S])
365         Z_met[np.isnan(Z_met)] = 0#convert nan to zero
366
367         #sys.exit(1)
368
369         #%%
370         #MCMC
371         model_met,x_pre,y_pre=met_model(x_samples,y_samples,z_samples,\
372                                         predict=True,x_pre=x_pre,y_pre=y_pre,\
373                                         sd_predict=sd_predict)
374         x_all_gps.append(x_pre)
375         y_all_gps.append(y_pre)
376         model_met.sample(iter=70000, burn=40000, thin=7)
377         print('logp after MCMC={:06.2f}'.format(model_met.logp))
378         sf_mean=sf.trace()[-500:].mean()
379         b_mean=b.trace()[-500:].mean()
380         b_s_mean=b_s.trace()[-500:].mean()
381         a_mean=a.trace()[-500:].mean()
382         a_s_mean=a_s.trace()[-500:].mean()
383         #updating the stopping
384         ev_chain,ev_mean,ev_std,ev_skew=ev_samples(b=b,b_s=b_s,sf=sf,down_lim=0, \
385                                                    up_lim=sigma_up)
386         stp_new=np.array([ev_mean,ev_std])
387         stp_stp.append(stp_new)
388         stopping=abs((stp_old-stp_new)/stp_new)
389         print "\nStopping = ",\
390         ["{:0.4f}".format(st) for st in stopping ],\
391         '\nPareto',"{:0.4f}".format(np.sqrt((stopping[0]/0.02)**2 + \
392                         (stopping[1]/0.1)**2))
393         print "mean =","{:0.3e}".format(ev_mean),\
394         "\tStd = ","{:0.3e}".format(ev_std),"\tSkew = ","{:0.3f}".format(ev_skew)
395         stp_old=stp_new
396 #     def met_model_fun(sf=sf_mean,a=a_mean,
397                               #a_s=a_s_mean,b=b_mean,b_s=b_s_mean,inpt=[1,1]):
398 #         return sf*norm.pdf(inpt[0],a,a_s)*norm.pdf(inpt[1],b,b_s)
```

```
399         SD_samples.append(model_met.trace('obs_met_predict')[:].std(axis=0))
400         sd_predict=model_met.trace('obs_met_predict')[:].std(axis=0)
401         all_predict.append(model_met.trace('obs_met_predict')[:])
402         add_sample_indx+=1
403         plt.figure(3)
404         plt.plot(x_samples,y_samples,'r.')
405         index_new=np.argmax(sd_predict)
406         new_y=y_pre[index_new]
407         new_x=x_pre[index_new]
408         New_z=np.empty(new_x.shape)
409         New_z[0]=true_model(new_x,new_y)
410         x_samples=np.concatenate([x_samples,new_x],axis=0)
411         y_samples=np.concatenate([y_samples,new_y],axis=0)
412         plt.plot(new_x,new_y,'b*')
413         plt.xlabel(r'$\theta$')
414         plt.ylabel(r'$\sigma$')
415         plt.xlim([theta_down,theta_up])
416         plt.ylim([0,sigma_up])
417         z_samples=np.concatenate([z_samples,New_z],axis=0)
418         name_fig_new_sample='sample_results/'+\
419         'fig_new_sample_GPS'+str(num_i)+'_'+'_'+str(add_sample_indx)+'.png'
420         plt.savefig(name_fig_new_sample)
421         plt.close('all')
422         if np.mod(add_sample_indx,50)==0:
423             name1='sample_results/summary_GPS'+str(num_i)+\
424             '_'+str(add_sample_indx)+'.txt'
425             name2='sample_results/SD_samples_GPS'+str(num_i)+\
426             '_'+str(add_sample_indx)+'.txt'
427             name3='sample_results/optimization_summary_GPS'+str(num_i)+\
428             '_'+str(add_sample_indx)+'.txt'
429             name4='sample_results/samples_new_'+'gaussian_GPS'+str(num_i)+\
430             '_'+str(add_sample_indx)+'.txt'
431             np.savetxt(name1, (logmap))
432             np.savetxt(name2, (SD_samples))
433             np.savetxt(name3, (x_best))
434             np.savetxt(name4, (x_samples,y_samples,z_samples))
435 #Save text data:
436 name1='sample_results/summary_GPS'+str(num_i)+'_'+str(add_sample_indx)+'.txt'
437 name2='sample_results/SD_samples_GPS'+str(num_i)+\
438 '_'+str(add_sample_indx)+'.txt'
439 name3='sample_results/optimization_summary_GPS'+\
```

```
440 str(num_i)+'_'+str(add_sample_indx)+'.txt'
441 name4='sample_results/samples_new_'+'gaussian_GPS'+\
442 str(num_i)+'_'+str(add_sample_indx)+'.txt'
443
444 np.savetxt(name1, (logmap))
445 np.savetxt(name2, (SD_samples))
446 np.savetxt(name3, (x_best))
447 np.savetxt(name4, (x_samples,y_samples,z_samples))
448 np.savetxt("sample_results/evidence_chain.txt",ev_chain)
449 np.savetxt("sample_results/a.txt",a.trace()[-500:])
450 np.savetxt("sample_results/a_s.txt",a_s.trace()[-500:])
451 np.savetxt("sample_results/b.txt",b.trace()[-500:])
452 np.savetxt("sample_results/b_s.txt",b_s.trace()[-500:])
453 np.savetxt("sample_results/sf.txt",sf.trace()[-500:])
454 np.savetxt("sample_results/sigma_like_met.txt",sigma_like_met.trace()[-500:])
455
456 #pm.Matplot.plot(model_met)
457 a_mean=a.trace()[-500:].mean()
458 a_s_mean=a_s.trace()[-500:].mean(axis=0)
459 b_mean=b.trace()[-500:].mean(axis=0)
460 b_s_mean=b_s.trace()[-500:].mean(axis=0)
461 sf_mean=sf.trace()[-500:].mean(axis=0)
462 z_met=meta_model(sf=sf_mean,a=a_mean,a_s=a_s_mean,b=b_mean,\
463                  b_s=b_s_mean,inpt=[T,S])
464 plt.figure(3)
465 #CS = plt.contour(x, y, z,cmap='Blues',levels=levels)
466 CS = plt.contour(T, S, EDF3,cmap='Blues')
467 CS = plt.contour(T, S, z_met,cmap='Reds',ls=".-")
468
469 plt.annotate('Metamodel from final samples', xy=(a_mean,\
470                                                  b_mean[0]+2*b_s_mean[0]), \
471 xytext=(a_mean+(theta_up-theta_down)/100., b_mean[0]+\
472         (sigma_up-sigma_down)/100.),\
473         arrowprops=dict(facecolor='red', shrink=0.05),)
474 plt.savefig('figures/models_from_final_samplesGPS'+\
475             str(num_i)+'_'+'_'+str(add_sample_indx)+'.png')
476 plt.savefig('figures/models_from_final_samplesGPS'+\
477             str(num_i)+'_'+'_'+str(add_sample_indx)+'.pdf')
478 #%%
479 plt.figure()
480 plt.plot(x_samples[:num_i],y_samples[:num_i],'ro')
```

```
481 plt.xlabel(r"$\theta$",fontsize=16)
482 plt.ylabel(r"$\sigma$",fontsize=16)
483 plt.xlim([theta_down,theta_up])
484 plt.ylim([0,sigma_up])
485 plt.contour(T, S, EDF3,cmap='Blues')
486 i=0
487 for sample in zip(x_samples[num_i:],y_samples[num_i:]):
488     plt.plot(sample[0],sample[1],'kD',ms=4)
489     i+=1
490     plt.text(sample[0]+(theta_up-theta_down)/100.,\
491             sample[1]+(sigma_up-sigma_down)/100.,str(i),fontsize=10)
492 plt.savefig('figures/added_samples_GPS'+str(num_i)+'_'+'.png')
493 plt.savefig('figures/added_samples_GPS'+str(num_i)+'_'+'.pdf')
494 #%%
495 plt.figure()
496 plt.plot(x_samples[:num_i],y_samples[:num_i],'ro')
497 plt.xlabel(r"$\theta$",fontsize=16)
498 plt.ylabel(r"$\sigma$",fontsize=16)
499 plt.xlim([theta_down,theta_up])
500 plt.ylim([0,sigma_up])
501 plt.contour(T, S, EDF3,cmap='Blues')
502 colors = [str(float(item)/(add_sample_indx)) for\
503         item in range(num_i,add_sample_indx)]
504 plt.scatter(x_samples[num_i:],y_samples[num_i:],c=colors,marker='D')
505 plt.savefig('figures/added_samples_color_GPS_'+str(num_i)+'_'+'.png')
506 plt.savefig('figures/added_samples_color_GPS_'+str(num_i)+'_'+'.pdf')
507 #%%
508 plt.figure()
509 for item in zip(x_all_gps,y_all_gps,\
510                 np.linspace(.2,1,add_sample_indx-num_i).tolist()):
511     plt.scatter(item[0],item[1],alpha=item[2],color='black',s=10)
512 plt.xlim([theta_down,theta_up])
513 plt.ylim([0,sigma_up])
514 plt.scatter(x_all_gps[-1],y_all_gps[-1],alpha=1.,color='red',s=10)
515 plt.savefig('figures/posterior_predictive_samples_GPS_'+str(num_i)+'_'+'.png')
516 plt.savefig('figures/posterior_predictive_samples_GPS_'+str(num_i)+'_'+'.pdf')
517 np.savetxt("sample_results/ALL_GPS_x.txt",x_all_gps)
518 np.savetxt("sample_results/ALL_GPS_y.txt",y_all_gps)
519 #%%
520 fig = plt.figure()
521 ax = fig.add_subplot(111, projection='3d')
```

```
522 x_mean=np.hstack([sf_mean,a_mean,a_s_mean,b_mean,b_s_mean])
523 Z_metamodel=met_model_fun(x=x_mean,inpt=[T,S])
524 ax.plot_surface(T,S,Z_metamodel,rstride=1,cstride=1,alpha=0.3,color='red')
525
526 for tick in ax.xaxis.get_major_ticks():
527     tick.label.set_fontsize(14)
528 for tick in ax.yaxis.get_major_ticks():
529     tick.label.set_fontsize(14)
530 for tick in ax.zaxis.get_major_ticks():
531     tick.label.set_fontsize(14)
532 ax.set_ylim([0,sigma_up])
533 ax.set_xlabel('\n'+r'$\theta$',fontsize=14)
534 ax.set_ylabel('\n'+r'$\sigma$',fontsize=14)
535 plt.savefig('figures/meta_model_3d.png')
536 plt.savefig('figures/meta_model_3d.pdf')
```